Pedro Miguel dos Santos Martins

Licenciado em Ciências da Engenharia Eletrotécnica e de Computadores

Monitoração ambiental em espaços fechados

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador:	José Manuel Fonseca, Professor Auxiliar com Agregação, FCT UNL
Co-orientador:	André Damas Mora, Professor Auxiliar, FCT UNL

Júri

Presidente:	Professor Auxiliar Doutor João Almeida das Rosas
Arguente:	Professor Auxiliar Doutor Tiago Oliveira Cardoso
Vogal:	Professor Auxiliar com Agregação Doutor José Manuel Fonseca

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2017

Monitoração ambiental em espaços fechados

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

AGRADECIMENTOS

Chegada a esta etapa da minha formação gostaria de agradecer todo o apoio que me foi oferecido por várias pessoas ao longo deste percurso, nomeadamente:

Ao professor José Manuel Fonseca, agradeço a transmissão de conhecimentos, a ajuda preciosa e o seu apoio que possibilitou a realização desta dissertação, enquanto Orientador.

Ao professor André Mora, agradeço a transmissão de conhecimentos, a ajuda preciosa e o seu apoio que possibilitou ultrapassar vários obstáculos durante a realização desta dissertação, enquanto Co-Orientador.

Aos meus pais Isabel Gregório e Policarpo Francisco, agradeço os seus incentivos, apoio, os bons conselhos e as oportunidades que permitiram levar esta tarefa a bom porto.

Ao meu irmão Luís Martins, agradeço a sua amizade, disponibilidade e conhecimentos transmitidos, durante o meu processo académico.

À Beatriz Mendes pelo seu apoio incondicional durante estes anos.

Aos meus colegas João Barata, Ricardo Martinho, Francisco Cruz, Pedro Maricato, Sérgio Coelho, Gonçalo Santos, João Santos, Sérgio Carrola e ao Fábio Lourenço pelos bons momentos, bons conselhos e **maus conselhos** que me ajudaram muito a chegar a esta fase da minha vida.

Aos meus amigos André Rezende e ao Ricardo Nogueira pelos bons momentos, bons conselhos e pelo suporte incondicional que me ajudou com que esta tarefa chegasse a bom porto.

RESUMO

A informação é imprescindível para analisar os sistemas, não basta, mas é imprescindível. Com novas informações é possível estudar e compreender melhor os sistemas em análise.

A obtenção de informação em qualidade e quantidade suficiente e garantir a sua salvaguarda é o objetivo principal desta dissertação.

A solução proposta é desenvolver uma arquitetura capaz de recolher informação sobre variáveis de temperatura, humidade, pressão atmosférica e concentração de gases nocivos. A qualidade da medida é assegurada pela calibração dos sensores envolvidos. A garantia da salvaguarda dos dados obtidos é realizada com o recurso ao armazenamento num cartão microSD.

Como resultado do objetivo proposto, é desenvolvida uma arquitetura específica que comporta uma rede de sensores "*Mesh*", capaz de sentir e recolher informação sobres as variáveis de interesse. Sempre que existirem dados em quantidade suficiente, uma aplicação "*Smartphone*"recolhe a informação disponível, através da tecnologia "*Bluetooth*", e armazena a mesma numa base de dados.

A arquitetura pode ser adaptada a qualquer espaço fechado e, a informação é disponibilizável para consulta num sítio web criado propositadamente para o efeito.

Palavras-chave: Recolha de informação; Quantidade; Qualidade de informação; Rede de sensores; Aplicação "*Smartphone*"; Base de Dados;"*Bluetooth*";

ABSTRACT

It is imperative to have information from a system in order to study and analyze it so that its faluts maybe fulfilled.

The main purpose of this dissertation is to gather quality information in a quantity such to be stored.

The purposed solution is to develop an arquitechture capable of collecting information about variables such as temperature, moisture, atmospheric pressure and harzadous gases concentration. Information quality is ensured through the used sensors calibration. Storage of gathered data is ensured via microSD card available capacity.

As a result of the purposed goal, a tailored made arquithecture was designed to hold a "*Mesh*"type sensor network, capable of congragate information about all variables of interest. Whenever information is avaliables in sufficent quantanty, a designed smartphone application shall collect it using Bluetooth technology and store it on a data base.

The designed architecture may be adapted to any environment, such as a mud technology house or an in vitro greenhouse. Resulting information is made available on a web site specifically created for this purpose.

Keywords: Collecting information; Quality information; Quantity; Sensor Network; Smartphone Application; Bluetooth ; Data Base storage;

ÍNDICE

Lista de Figuras

LISTA DE TABELAS

LISTAGENS

CAPÍTULO 1

INTRODUÇÃO

"Informação é poder"é uma frase que representa um dos conceitos mais antigos da humanidade. Este poder está presente nas mais diversas formas, em cada segundo do nosso quotidiano.

O como "fazer fogo", a observação das fases da lua e a sua relação com as marés, a observação do funcionamento de um enxame de abelhas, da circulação sanguínea, a análise de um percurso de um salto em altura de um atleta e até como funciona o movimento dos olhos de um leitor numa página web, são exemplos da importância da "observação objetiva"e do registo de dados. Estas ações permitiram a construção do conhecimento ao longo dos tempos.

Como se sabe para evoluirmos é fundamental obter dados, não basta, mas é imprescindível. O conhecimento é informação. A informação, genericamente, assenta na sistematização e correlação de dados recolhidos sobre determinado objeto em análise.

A obtenção de dados de qualidade e em quantidade suficiente não é tarefa trivial, dado que consome muitos recursos de um projeto, a que está associado um custo muito elevado.

A presente dissertação apresenta uma arquitetura de monitorização ambiental em espaços fechados, de baixo custo energético, robusta, sem custos de telecomunicações e flexível no ambiente em que se insere.

Esta arquitetura de monitorização foi dividida em três fases. A primeira pretende interligar vários sensores em rede usando a tecnologia "*Mesh*". A segunda pretende a preparação de um servidor com uma base de dados e uma interface gráfica que permite o registo de dados e a visualização dos mesmos, na forma pretendida. Por fim a última fase, pretende o desenvolvimento de uma aplicação "*Bluetooth*", para "*Smartphone*", que recolha os dados da rede de sensores e mais tarde possa envia-los para o servidor, logo que haja uma ligação à "*Internet*".

Esta arquitetura vai ser aplicada na recolha de dados do efeito térmico numa casa construída com tecnologia de lama, pelo departamento de Engenharia Civil, sediada no campus da Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa. Esta aplicação pode ser ajustada para a recolha de dados térmicos numa estufa, para estudo do impacto das diferenças térmicas no crescimento das plantas. Há uma empresa Portuguesa que poderá vir a manifestar interesse neste âmbito.

1.1 Motivação

Apesar de existirem vários sistemas de monitorização, a arquitetura proposta permite adaptar o sistema à realidade de forma a otimizar a recolha de dados e garantir que estes são registados, para posterior análise. A análise desses dados sentidos pelos sensores poderão colmatar as falhas dos sistemas.

1.1.1 Aplicações

Para aplicar esta arquitetura num sistema real foram feitas duas propostas. A primeira proposta foi colocada pela Professora Paulina París, professora do departamento de Engenharia Civil, responsável pela construção de uma casa utilizando técnicas à base de lama. Este projeto pretende estudar o efeito térmico neste tipo de construção de casas para posterior comparação com uma casa de construção contemporânea. Outra alternativa para esta aplicação poderá ser para uma empresa Portuguesa, especializada na reprodução de plantas pelo método in vitro. Esta empresa detém várias estufas em Portugal e no estrangeiro e ao comparar os resultados das diversas unidades de produção, detetou discrepâncias nos resultados. A estufa em Portugal apresenta algumas características interessantes:

- Variações térmicas até 3 [° C] ao longo da estufa.
- Normalmente existe um termómetro por estufa, pelo que só regista a temperatura num ponto da estrutura.
- Além destas duas situações, poderão existir outras variáveis que afetam o desenvolvimentos das plantas.

No caso da "casa com tecnologia de lama", será interessante registar dados sobre a temperatura, humidade e outros componentes do ar, para mais tarde se poder comparar com as casas de construção contemporânea. Na aplicação alternativa será interessante adaptar a mesma arquitetura para a monitorização cuidada dos vários locais estrategicamente selecionados nas estufas, onde se poderá analisar o comportamento das diversas variáveis que a aplicação permite registar, tanto no solo como no ar.

O objetivo é recolher os dados necessários dos vários pontos das salas de cultura, para correlacionar as condições ambientais com o desenvolvimento das plantas.

Uma das vantagens que esta arquitetura também apresenta a possibilidade de facilmente adicionar sensores, que permitem a monitorização de outras variáveis, tornando esta numa designada arquitetura de monitorização modular.

CAPÍTULO 2

ESTADO DA ARTE

Neste capítulo pretende-se apresentar projetos académicos e comerciais sobre sistemas de monitorização e, de controlo de estufas e uma síntese do estado da arte.

Nos projetos académicos pretende-se aprofundar quais são os objetivos principais, quais os resultados obtidos quando foram realizados testes e as conclusões mais relevantes. No caso dos projetos comercias pretende-se aprofundar quais as suas áreas de negócios e alguns produtos que possam ser relevantes para o desenvolvimento desta dissertação.

Após a análise dos projetos anteriormente indicados será realizada uma síntese do estado arte que pretende aprofundar as tendências da área, os seus impactos, questões, problemas em aberto e as possíveis soluções encontradas.

Por fim pretende-se apresentar um quadro síntese das tecnologias usadas nos projetos académicos.

2.1 Projetos Académicos

2.1.1 Multi-monitorização de Estufa Agrícola

Este projeto consiste num sistema com a capacidade de monitorizar e controlar as variáveis de temperatura, humidade de uma estufa. Os objetivos deste projeto eram melhorar a fiabilidade e eficácia da estufa, reduzir a carga de trabalho para o produtor, rentabilizar os recursos energéticos e garantir padrões de qualidade dos produtos finais.

O sistema é composto por diversos sensores para monitorizar as diversas variáveis e por um computador que garante a consistência dos dados, através do uso de uma Base de Dados em SQL. Para complementar o sistema este tem a capacidade de comunicar o estado da estufa através do serviço "*Short Message Service*"(SMS), "*Wireless Application Protocol*"(WPA) e um site.

No entanto o sistema ainda não está completo, falta concluir o desenvolvimento do sistema de monitorização e o serviço SMS. Devido a dificuldades financeiras, não foram capazes de obter um computador para este efeito. Como consequência não foram obtidos resultados.

Este projeto foi desenvolvido em parceria com a Escola Superior de Tecnologia, Escola Superior Agrária e o Instituto Politécnico de Castelo Branco, e o seus autores são respetivamente J. Morato, S. Cruz, F. Pereira e J. C. Metrôlho. Para mais informação detalhada consulte o artigo [33].

2.1.2 Monitorização de uma Estufa através de uma Rede de Sensores

O artigo "*Greenhouse Monitoring with Wireless Sensor Network*"pretende apresentar um sistema, através de uma rede de sensores, que permite monitorizar a temperatura, humidade, luz e dióxido de carbono de uma estufa. Este sistema é composto por vários rádios com os sensores, que comunica os dados recolhidos para um "*gateway*". O sistema foi montado e testado na estufa "*Martens Greenhouse Research Foundation's*"na Finlândia do qual foi verificado que o sistema pode monitorizar vários pontos da estufa utilizando um número reduzido de aparelhos de medição, existiam dificuldades na comunicação devido à elevada concentração de humidade e pela mesma razão os aparelhos de medição danificaram-se. A tecnologia de comunicação usada foi o "*ZigBee*".

Este projeto foi realizado por Teemu Ahonen, Reino Virrankoski e Mohammed Elmusrati na "*University of Vaasa*"no "*Department of Computer Science*"em "*Vaasa*"na Finlândia. Consulte o artigo [2] para mais informações.

2.1.3 Sistema de Controlo e Monitorização de uma Estufa através do ZigBee

O artigo "*Grennhouse Monitoring and Control System Based on ZigBee*"pretende apresentar um sistema de monitorização e um sistema de controle para estufas. As variáveis monitorizadas foram a temperatura, humidade, dióxido de carbono e a humidade no solo. O objetivo deste projeto foi tornar o setor agrícola mais competitivo, apresentado produtos de maior qualidade e em maior quantidade.

Apenas o sistema de monitorização foi concluído, no entanto não é apresentado resultados sobre este sistema. A informação que irá ser recolhida irá ser analisada para encontrar a melhor solução para o sistema de controle.

Este projeto foi realizado por: Minghun Shang, Guoying Tian,Leilei Quin,Jia Zhao, Huaijun Ruan, Fengyun Wang da "*S&T Information Engineering Research Center, Shandong Academy of Agricultural Sciences,Jinan, China*"e da "*Shandong Product Quality Supervision & Inspection Research Institute, Jinan, China*", para mais informações consulte o artigo [48].

2.1.4 Sistema de Monitorização de Estufas através de uma Rede de Sensores Wireless

O artigo "*Greenhouse Monitoring System Based on a Wireless Sensor Network*"pretende apresentar um sistema de monitorização de estufa através da placa de comunicação "*ZigBee*".Para complementar o sistema, esta garante a consistência dos dados através de uma Base de Dados e um site para consulta dos dados.

Os objetivos deste projeto são, utilizar a WSN, provar que não existiam variações no micro-clima gerado pela estufa.

O projeto foi testado numa estufa com uma área 160 m^2, com os resultados obtidos concluiu-se que existiam variações do micro-clima ao longo da estufa, e foi possível construir um sistema de monitorização usando como recurso as WSN.

Este projeto foi desenvolvido por: Ilias Lamprinos, Marios Charalambides e Michael Chouchoulis no "*Telco Software Department, INTRACOM TELECOM*"[22].

2.1.5 Desenho de um sistema de monitorização remoto usando a plataforma Android

O artigo *Design of Greenhouse Enviroment Remote Monitoring System Based on Android Platform* pretende apresentar um sistema de monitorização de estufas combinando a tecnologia de comunicação 3G e ZigBee, a aplicação "*Android*"permite controlar e monitorizar a estufa.

O sistema de monitorização e a aplicação estão conclui-dos, o sistema em causa melhora a forma como o micro-clima é mantido dentro da estufa e, promove as "*Smart Greenhouse*"e o "*IOT*".

Este projeto foi desenvolvido por Li Zhang, Congcong Li, Yushen Jia, Zhigang Xiao do "*College of Mechanical & Electrical Engineering*"e da "*Agricultural University of Hebei*"na China [18].

2.1.6 Sistema de Controlo e Monitorização Wireless para uma estufa

O artigo "*Wireless Monitor and Control System for Greenhouse*"pretende apresentar um sistema de monitorização e controlo de uma estufa utilizando as tecnologias "*ZigBee*"e "*Wireless Sensor Network*". As variáveis a monitorizar são a temperatura, humidade, intensidade de luz.

O sistema foi testado numa caixa fechada, com o propósito de simular uma estufa, a caixa tinha as seguintes dimensões 50x30x60 cm^2 e foram recolhidos dados durante um período de 16 horas. Foram realizados dois testes.

- 1ºTeste: Permitiu testar o sistema de monitorização e os dados recolhidos serviram para construir a funções de transferência dos controladores. Foi observado que a temperatura aumentava com a exposição solar.

- 2º Teste: Foi adicionado à caixa uma ventoinha que permitiu controlar as temperatura e a humidade. Foi observado que os valores de temperatura e humidade tornaram-se mais acentuados com o funcionamento da ventoinha.

Foi conclui-do que o sistema foi construido com êxito e foi possível recolher dados sobre as variáveis em questão. Este projeto foi desenvolvido pelo Rana H. Hussain, esta informação encontra-se "*International Journal of Computer Science and Mobile Computing, Vol.2 Issue. 12, December- 2013 na pág. 69-87*", para mais informações consulte o artigo [25].

2.1.7 Monitoração Wireless de uma estufa usando controladores

Este artigo *Wireless Monitoring of the Green House System Using Embedded Controllers* pretende apresentar um sistema de monitorização e de controlo das variáveis de temperatura, humidade e, luz para mais tarde estudar o seu impacto na qualidade de produção de uma estufa. O objetivo principal é encontrar uma solução para aumentar a exportação de rosas, mantendo a qualidade das mesmas, em Nasik.

Foram realizados dois tipos de testes em áreas distintas da estufa, uma para o sistema de monitorização, que permitiu concluir que o sistema estava a operar nas condições ideais e, a segunda área para o sistema de controlo, que permitiu observar o controlo das variáveis de temperatura e de humidade encontrando a melhor relação para uma melhor produção das rosas.

Com este projeto foi possível verificar que a solução encontrada é barata, pouca manutenção é necessária, flexibilidade da zona a monitorizar e o mesmo sistema pode ser usado em ambientes industriais.

Este projeto foi desenvolvido por Miss.Vrushali R. Deore e por Prof. V.M. Umale [16].

2.1.8 Solução de uma rede de sensores para a agricultura de precisão usando a tecnologia ZigBee

Este artigo "*A Wireless Sensor Network Solution for Precision Agriculture Based on ZigBee Technology*"pretende apresentar um sistema de rede de sensores (WSN) usado para a agricultura de precisão. Este é capaz de recolher a informação do ambiente, analisar os novos dados para que as variáveis de monitorização possam ser controladas em tempo real. Esta rede de sensores segue o standard IEEE 802.15.4.

Os teste realizados foram apenas para testar as várias topologias de comunicações entre "*Nodes*"em que a topologia proposta é que demonstra a melhor performance. Com este projeto foi verificado que as WSN são uma boa solução para aplicar a agricultura de precisão, melhorando significativamente a produção e a qualidade das plantações.

2.2 Projetos Comerciais

2.2.1 *Monnit Corporation*

A "*Monnit Corporation*"é uma empresa que comercializa produtos, onde tenta integrar a monitorização e o controlo de sistemas com as IOT. A gama de produtos desta empresa vai desde a venda dos sensores até ao sistema de integração para a monitorização e controlo do local de interesse.

Os sensores são wireless e comunicam com um "*Wireless Gateway*". Este último porventura poderá estabelecer a ligação à "*Internet*"através do 2G/3G, ou "*Wireless Usb*", ou "*Ethernet*"e "*Serial*".

Para além da comercialização de hardware, a "*Monnit Corporation*"em complemento da sua actividade vende aplicações específicas e adaptadas para cada sector de atividade na indústria, como por exemplo a monitorização de edifícios, restaurantes, hotéis, etc. Informe-se no site [21].

2.2.2 Sensaphone

Desde de 1985 que a "*Sensaphone*"cria soluções para o mercado de monitorização de sistemas remotos. Permite ainda notificações desenvolvidas à medida e alarmes prévios, para que os problemas possam ser resolvidos com antecedência. Os seus produtos têm a capacidade para ler vários sensores em simultâneo e como incluí o uso de baterias torna o sistema mais independente. As comunicações podem ser feitas através da linha do telefone, ou 2G/3G, ou "*Ethernet*". Tudo isto sem custos adicionais.

Têm como clientes as áreas de tratamento de águas residuais,petróleo e gás natural, Data Centers, medicina, residências, etc. Consulte o site [46].

2.2.3 Libelium

A "*Libelium*"é uma empresa Espanhola sediada em "*Zaragoza*", em que o seu negócio consiste no desenvolvimento de sistemas de monitorização para diversos ambientes. Já tem projetos desenvolvidos nas área da "*smart cities*, "*smart grid*", "*smart agriculture*", etc. Tem como clientes, a "*Nasa*", "*Telefónica*", "*Philips*", "*IBM*", "*Vodafone*", e muitos mais. Como produtos, tem o "*waspmote*"onde os diversos sensores são ligados e comunicam com um "*gateway*", denominado por "*Meshlium*". Este equipamento posteriormente faz a ligação à internet via "*Wi-Fi*", 2G/3G, ou ainda por "*ethernet*". Ainda vende um equipamento que permite a leitura de dados de sensores e o envia dos mesmos pela a "*Internet*"via "*Wi-Fi*", denominado "*Plug and Sense*". A informção exposta está de acordo com o sítio [28].

2.2.4 Argus Control Systems Ltd

A empresa "*Argus Control Systems Ltd*"fornece sistemas automáticos de controlo para a horticultura, aquacultura, estufas e para indústrias ligadas às biotecnologias. O sistema

permite controlar e monitorizar a temperatura e humidade do ar, a temperatura da água, etc. A empresa tem vastos conhecimentos em análise de períodos de luz, de irrigação avançada e vasta experiência na aplicação dos nutrientes que as plantas necessitam.

No caso específico das estufas, os tipos de plantações podem ser vários, tais como ,as de vegetais, de flores, de propagação/reprodução, ou de produtos medicinais, etc. A empresa fornece o equipamento necessário para o controlo e monitorização de sistemas e ainda o respectivo software. O sistema é composto por os módulos I/O, onde são ligados os sensores. De seguida os módulos I/O comunicam com os controladores de rede, que por sua vez enviam os dados para um servidor. Mais tarde os clientes podem aceder aos dados do servidor através da sua rede ou por acesso remoto. Consulte o sitio [10].

2.2.5 *Sensohive*

A "*Sensohive*"é uma empresa sediada na Dinamarca e foi fundada por Tim Casper e por Tobias Ejersb. O objetivo da empresa é simplificar o mercado das IOT e torná-lo mais acessível. Fornecem aos seus clientes sensores "*Wireless*"utilizando a sua rede de IOT ou a "*cloud platform*"[47].

A empresa tem alguns prémios tais como, "*EnergiFyn*", "*Dansk Industri*", "*World Cup*", etc.

A sua gama de produtos vão desde sensores wireless como estações bases e "*gateways*"que permite tornar nos seus produtos em produtos IOT. A aplicação "*Sensohive*"disponível para computador, "*Tablet*"e "*Smartphone*", permite analisar a informação, monitorizar, armazenar em "*cloud*"e alarmes por SMS ou por e-mail. Permite ainda controlar e adaptar a apresentação da informação por gráficos, mapas, alarmes, exportação para ficheiros e integração com outros softwares.

2.2.6 *Climate Control Systems INC*

A "*Climate Control Systems Inc*"é uma empresa que tem vindo a construir sistemas de automação desde 1985. Os seus principais produtos são "*Climate Manager*", "*Fertigation Manager*"e o "*Fertigation Water Treatment*". Estas tecnologias são aplicáveis à reciclagem de água, à redução de custos de energia, promover soluções comerciais para estufas, etc. O sistema "*Climate Manager*", permite controlar algumas variáveis ambientais e incluí ainda um sistema de irrigação. O produto "*Fertigation Manager*"é um sistema de controlo para estufas com flores e vegetais que monitoriza o fluxo de água, condutividade, pH e ainda um fertilizador selecionado. Consulte o site [15].

2.2.7 *Lord Microstrain Sensing Systems*

A "*Lord Microstrain Sensing Systems*"é uma empresa sediada na Carolina do Norte nos Estados Unidos da América, fundada em 1987. Produzem sensores e sistemas com uma ampla variedade de aplicações, que vão desde área Aeroespacial, à Engenharia Civil, à

Produção Industrial , Petróleo e Gás, Robótica, etc. O sistema é composto pelos sensores que comunicam com um "*router*"que por sua vez permite conexões de vários dispositivos. A comunicação pode ser feita através de satélite, "*ethernet*", "*usb*", "*gateway*", etc. Tem ainda uma aplicação para computador e soluções "*cloud*"[57].

2.3 Síntese do Estado da Arte

Os artigos, apresentados na secção "Projetos Académicos"presente neste capítulo 2.1, transmitem a ideia de que as estufas são elementos vitais para o desenvolvimento de uma agricultura de precisão e mias moderna, no entanto este último, pode ser mais eficaz com a ajuda de sistemas de monitorização e de controlo. Estes sistemas aumentam a capacidade de produção e os padrões de qualidade são mais refinados.

Os projetos comerciais, apresentados na secção "Projetos Comerciais"presentes neste capítulo 2.2, apresentam diversos produtos de monitorização e controlo de estufas, em que estes últimos, podem estar ligados à "*internet*", tornando os seus projetos em "Projetos de IOT". A grande maioria destes projetos comerciais são complementados por aplicações para "*SmartPhone*", Web e para computador que permitem de alguma forma monitorizar e controlar as estufas. A grande vantagem destas aplicações é a possibilidade de alarmes que possibilita o aviso de problemas em tempo real ao utilizador.

Ao analisar os projetos num conjunto só, podemos afirmar de que a tecnologia de comunicação mais popular é a tecnologia Wireless, usando como recurso o conceito "*Wireless Sensor Network*". Este conceito como permite que existam vários aparelhos de medição ligados entre si, foram desenvolvidas topologias de comunicação baseadas no standard IEEE 802.10.15, ou seja "*Mesh*". As grandes vantagens de utilização deste standard é que os aparelhos de medição podem ser desenvolvidos com um número específico de sensores, mas o sistema pode ser facilmente escalável com outros sensores e com um uso reduzido de aparelhos de medição podem ser recolhidos dados de diversos locais.

Um dos problemas que foi identificado, mas que não encontraram a solução, é quando dentro da estufa existe uma elevada concentração de humidade, torna a comunicação lenta e pouco fiável, perdendo dados de comunicação. Nestes casos em especifico não existia a garantia da consistência dos dados recolhidos.

2.4 Conclusão

Ao analisar com detalhe o estado da arte e ao estudar cada hipótese do tipo de tecnologias de comunicação existentes, qual o tipo de aplicações que os sistemas de monitorização podem ter é possível afirmar que a solução que vai ser implementada nesta dissertação será usar o standard IEEE 802.10.11. Este standard permite criar uma rede Mesh em que vários rádios podem estar ligados na mesma rede. Estes rádios podem ter capacidades de monitorização com diversos sensores montados, tornando-os assim em aparelhos

de medição, isto é, cada "*Node*"ou "*Hop*"tem capacidade de medir ou sentir variáveis de interesse consoante o sensor ligado.

Apesar de grande maioria das soluções apresentadas terem uma comunicação à internet para comunicar os dados recolhidos para uma base de dados, nesta dissertação, que irá ser desenvolvida uma arquitetura, não vai ter uma ligação à internet a partir de um computador. Ou seja, um "*Smartphone*"com a capacidade de se ligar à internet vai servir de ponte para a comunicação dos dados até à Base de Dados.

Grande maioria das Bases de Dados apresentados foram desenvolvidas em SQL, visto que é a solução popular, nesta dissertação também vai ser desenvolvida uma base de dados em SQL.

CAPÍTULO 3

FUNDAMENTOS TECNOLÓGICOS

3.1 Micro-Controlador Pinoccio

O micro-controlador "*Pinoccio*"é constituído por um processador "*Atmel - AT-Mega256RF2*"inclui um rádio "*built-in*", uma bateria de 550 mAh, 17 portas digitais com PWM, 8 portas analógicas, dois circuitos UART, um interface com o protocolo SPI, um interface com o protocolo I_2C e uma porta micro USB para alimentação e carregamento de programas. A programação é realizada na aplicação "*Arduino IDE*", apesar deste micro-controlador não ser desta marca, esta aplicação permite o carregamento do programa para o micro-controlador "*Pinoccio*". A Tabela 3.1 apresenta as especificações técnicas, a informação está disponível no sitio [38].

Tabela 3.1: Especificações do micro-controlador "*Pinoccio*"

Micro-controlador	"*AT−Mega256RF2*"
Tensão de Funcionamento [V]	1.8a3.3
Terminais Digitais I/O	17
Terminais PWM Digitais I/O	17
Terminais Analógicos I	8
Flash Memory [KB]	256
SRAM [KB]	32
EEPROM [KB]	8
Velocidade de Clock [MHz]	16
Comprimento [mm]	53
Largura [mm]	25
Peso [g]	12

3.2 Micro-Controlador Arduino

O projeto "*Arduino*", desenvolvido em Itália em 2005, é um projeto "*Open Source*"que trouxe vários micro-controladores "*Arduino*", tais como o "*Arduino Uno*", "*Arduino Nano*", "*Arduino Lillypad*", "*Arduino Mega*"e muitos mais. Infromação disponibilizada no site [3]. Nesta dissertação vão ser utilizados o "*Arduino Uno*"e o "*Arduino Nano*". Estes micro-controladores são placas de prototipagem rápida e são constituídos na sua grande maioria por um processador "$AT-Mega$"e por uma EEPROM. O processador "$AT-Mega$"realiza a computação e a EEPROM permite o carregamento de programas para estes possam ser executados pelo processador.

A linguagem de programação é o C/C++, no entanto existem vários IDE que possibiliatam programar e carregar o programa para o micro-controlador "*Arduino*", como por exemplo o "*Arduino - plugin*"para o IDE "*Netbeans*"[4] , "Arduino IDE for Visual Studio"para o IDE "*Visual Studio*"[6] e até um site da "*Autodesk*"que permite emular o micro-controlador "*Arduino*"e programa-lo [12]. Nesta dissertação será usado "Arduino IDE", disponibilizado em [5].

3.2.1 Especificações do modelo Arduino Uno

O "*Arduino Uno*"é um micro-controlador baseado no micro-processador "$AT-mega328P$", com 14 portas digitais de "*Input/Output*"das quais 6 portas podem ser usadas como PWM "*output*"e ainda com 6 portas analógicas de "*input*". Este micro-controlador contém também um cristal de quartzo de 16 [MHz], uma porta USB e um porta de alimentação [9, 36].

O micro-controlador "*Arduino Uno Rev3*", que é a terceira e a mais recente versão do modelo "*Arduino Uno*", tem duas interfaces para o protocolo de comunicação I_2C. Nas outras versões do micro-controlador "*Arduino Uno*, "*Arduino Uno Rev 1*"e o "*Arduino Uno Rev 2*", existe apenas uma interface I_2C. O protocolo de comunicação mais usado com os dispositivos periféricos ao micro-controlador "*Arduino Uno*"é o "*Serial Protocol Interface*"(SPI), informação mais detalhada encontra-se neste capítulo no secção "*Serial Protocol Interface* (SPI)"3.2.3.1.

Na Tabela 3.2 vai ser apresentado as especificações técnicas do micro-controlador "*Arduino Uno Rev3*".

Tabela 3.2: Especificações do micro-controlador "*Arduino Uno Rev*"3

Micro-controlador	"*AT−Mega 328P*"
Tensão de Funcionamento [V]	5
Tensão de Entrada recomendada [V]	7 a 12
Tensão de Entrada limite [V]	6 a 20
Terminais Digitais I/O	14
Terminais PWM Digitais I/O	6
Terminais Analógicos I	6
Corrente DC por terminal [mA]	20
Corrente DC por terminal 3.3V [mA]	50
Flash Memory	32[KB] dos quais 0.5 [KB] é o bootloader
SRAM [KB]	2
EEPROM [KB]	1
Velocidade de Clock [MHz]	16
Comprimento [mm]	68.66
Largura [mm]	53.4
Peso [g]	25

3.2.2 Especificações do modelo Arduino Nano

O micro-controlador "*Arduuino Nano*"tem um micro-processador "*AT−mega328*", 22 portas digitais "*Input/Output*"das quais 6 são PWM "*output*"e 8 portas analógicas "*Input/Output*". Contém ainda um cristal de quartzo a 16 [MHz] e uma porta USB do tipo "*micro usb*". No entanto, este modelo não contém nenhuma porta de alimentação externa como a do "*Arduino Uno Rev 3*".

O micro-controlador "*Arduuino Nano*"tem especificações semelhante ao dp "*Arduino Uno Rev 3*", no entanto difere pelo tamanho compacto e reduzido do micro-controlador "*Arduuino Nano*"[29].

A Tabela 3.3 apresenta as especificações técnicas do micro-controlador "*Arduino Nano*".

Tabela 3.3: Especificações do micro-controlador "*Arduino Nano*"

Micro-controlador	"*AT−Mega 328*"
Arquitetura	AVR
Tensão de Funcionamento [V]	5
Tensão de Entrada recomendada [V]	7 a 12
Terminais Digitais I/O	22
Terminais PWM Digitais I/O	6
Terminais Analógicos I	8
Corrente DC por terminal [mA]	40
Flash Memory	32 KB dos quais 2 KB é o bootloader
SRAM [KB]	2
EEPROM [KB]	1
Velocidade de Clock [MHz]	16
Comprimento [mm]	45
Largura [mm]	18
Peso [g]	7

3.2.3 Protocolos de comunicação

Os micro-controladores "*Arduino Uno*"3.2.1 e "*Arduino Nano*"3.2.2 foram desenhados para possibilitar a integração de vários dispositivos periféricos. Estes dispositivos podem ser sensores, módulos de comunicação, leitores de cartões "*Secure Digital*"(SD), "*Real Time Clock*", motores, ecrãs, etc. Mas para que esta integração seja possível são utilizados dois protocolos de comunicação, o "*Serial Protocol Interface*"(SPI) e o "*Inter Integrated Circuit*"($\boldsymbol{I_2C}$).

Por motivos de teste, o micro-controlador "*Arduino*"também pode comunicar com um computador via comunicação série, no entanto este protocolo de comunicação assenta sob o "*HardwareUniversal Asynchrounous Receiver Transmitter*"(UART).

Os próximos quatro temas irão apresentar as caraterísticas de cada protocolo de comunicação e o hardware UART.

3.2.3.1 "*Serial Protocol Interface*"(SPI)

O protocolo de comunicação de dados "*Serial Protocol Interface*", mais conhecido por "SPI", é uma interface síncrona, com ligação em série, que usa uma topologia "Master and Slave"[7, 11].

Este protocolo de comunicação é utilizado em dispositivos, tais como, "*Cartões Secure Digital*"(SD), ecrãs de cristal, sensores, câmaras e "*Memory Flash EEPROM*".

O protocolo "SPI"é composto por quatro terminais conforme descrição na Tabela 3.4

Na Tabela 3.4, o terminal "MISO - Master In Slave Out"significa que os dados saem do "Slave"para o "Master", no terminal "MOSI- Master Out Slave In"tem a função contrária ao

Tabela 3.4: Descrição dos terminais do protocolo de comunicação SPI

Terminal	Descrição
MISO	"Master in Slave Out"
MOSI	"Master Out Slave In"
SCK	"Serial Clock"
SS	"Slave Select"

terminal "MISO", isto é, os dados saem do "Slave"para o "Master", o terminal "SCK - Serial Clock"o "Master"fornece o sinal de "Clock"para "Slave"e por fim o terminal "SS - Slave Select", também conhecido por "CS - Chip Select"tem como função ativar ou desativar o dispositivo "Slave".

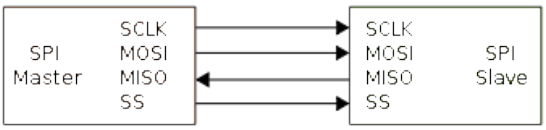

Figura 3.1: Exemplo de ligação "SPI"entre dois dispositivos

3.2.3.2 "*Inter Integrated Circuit*"(I_2C)

O protocolo de comunicação de dados "Inter Integrated Circuit", desenvolvido pela Philips é utilizado para conectar vários dispositivos de velocidade reduzida a uma placa-mãe e utiliza a topologia "Master and Slave". O modelo I_2C é constituído por dois canais de comunicação bidirecional. As tensões mais utilizadas são 5 [V] e 3.3 [V], no entanto é permitido utilizar outros valores de tensão. O modelo de referência I_2C ocupa 7 bit a 10 bit de espaço de endereçamento o que permite a existência de vários modos de velocidade tais como os apresentado na Tabela 3.5. Informação exposta em [11, 45].

Tabela 3.5: Modos de Velocidade do protocolo de comunicação I_2C

Modo	Velocidade	Unidade
Modo Padrão	100	[Kb/s]
Modo Velocidade Reduzida	10	[Kb/s]
"Fast Mode"na Versão 1 do I_2C	400	[Kb/s]
"Fast Mode"na Versão 3 do I_2C	1	[Mb/s]
"Hight Speed Mode"na Versão 2 I_2C	.4	[Mb/s]

Na Tabela 3.5 o "Modo de Velocidade Reduzida"propicia que a velocidade de comunicação seja inferior à velocidade do "Modo Padrão", no entanto a frequência de "*Clock*"também tem de ser inferior à frequência de "Clock"no Modo Padrão. Este protocolo facilita interligação de vários dispositivos ao mesmo canal como se pode observar

na Figura 3.2.

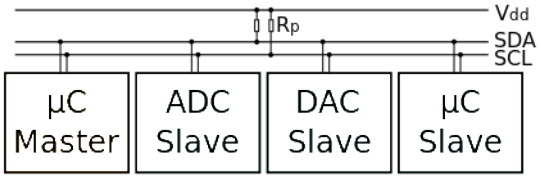

Figura 3.2: Exemplo de ligação I_2C entre vários dispositivos

Como podemos observar na Figura 3.2, existem 2 canais bidirecionais, o "SDA"e o "SCL". O canal bidirecional "*Serial Data*"(SDA) tem como função transmitir os dados, já o canal "*Serial Clock*"(SCL) tem como função de transmitir o sinal de "*Clock*", este canal é constituído por resistências "*PullUp*".

O Modelo Padrão é constituído por dois canais de comunicação, com endereçamento de 7 bit. Têm funções definidas e sabem qual é a função de um dispositivo "*Master*"ou de um "*Slave*".

O dispositivo "*Master*"fornece aos vários dispositivos "*Slave*"o sinal de "*Clock*"e inicia a comunicação. Já o dispositivo "*Slave*"recebe o sinal de "*Clock*"e espera que seja endereçado pelo "*Master*". É possível a existência de outros dispositivos "*Master*"na mesma rede I_2C, utilizando o protocolo "*MultiMaster*".

3.2.3.3 Comunicação Série

A comunicação série normalmente é utilizada entre o micro-controlador "*Arduino*"e um computador, além de viabilizar a comunicação com outros dispositivos, como é o caso de dois micro-controladores a comunicarem entre si. Para estabelecer a comunicação entre estes dois dispositivos, através da comunicação série, é necessário em primeiro lugar fazer as ligações físicas seguindo-se a gestão da respetiva comunicação. A Tabela 3.6 representa a forma como se deve ligar dois dispositivos usando a comunicação série.

Tabela 3.6: Ligações entre dois dispositivos usando a comunicação série

Dispositivo	**Dispositivo 2**
Tx	Rx
Rx	Tx

O micro-controlador "Arduino"também utiliza a comunicação série via usb, porém a comunicação via terminais RX/TX não funciona em conjunto com a comunicação via usb [8]. Consulte o capítulo Anexo na secção Comunicação Série I.2.1.2 para aprofundar o modo como a comunicação série é gerida via "*software*".

3.2.3.4 "*Universal Asynchrounous Receiver Transmitter*"(UART)

O hardware "*Universal Ansychronous Receiver Transmitter*"(UART) é um dispositivo que permite a comunicação série entre duas máquinas.

A forma como a informação é encapsulada e enviada faz-se da seguinte forma. Em primeiro lugar é enviado uma caracter de cada vez de uma palavra. O caracter pode ocupar entre 5 a 9 [bit]. Para iniciar a comunicação é enviado um bit de inicio que fisicamente significa ter a tensão a 0 [V] na linha. Após o envio do bit de inicio e dos bits correspondentes ao caracter é enviado um bit de finalização, que fisicamente significa colocar a tensão máxima na linha.

A parte correspondente do hardware que trata da receção é controlado por um sinal de "*clock*", esse hardware necessita de operar 8 vezes mais rápido que o sinal de "*clock*". A operação de receção é dividida em duas fases. A primeira fase a cada impulso de "*clock*", o hardware verifica a tensão na linha. Caso a tensão na linha esteja a 0 [V] significa que se iniciou a transmissão de um caracter. A segunda fase é avaliar o tempo de receção, caso este seja metade do tempo de bit, o hardware acusa o início de receção da informação. O novo caracter é guardado num "*shift register*".

A operação de transmissão só necessita de colocar o caracter a enviar para o "*shift register*"e o hardware inícia automaticamente a comunicação [54].

3.3 Características de sensores

As características dos sensores é um conjunto de conceitos que avaliam a qualidade dos mesmos e o que influencia a sua seleção. Os conceitos são vários, no entanto os de maior relevância são:

1. Função de transferência: A função de transferência é a razão entre o sinal de saída e o sinal de entrada [19];

2. Precisão: A precisão tem como base o desvio-padrão de uma série de resultados da mesma análise. A precisão depende da fiabilidade e da repetibilidade do sensor. No entanto fatores como a temperatura, pressão, vibração, aceleração e ruído eletromagnético afetam o seu nível de precisão [19];

3. Gama de entrada: Intervalo de valores que o sensor tem capacidade de converter [19];

4. Gama de saída: Intervalo entre o valor mínimo de saída e o valor máximo de saída [19];

3.4 Sensores

Os sensores são úteis para sentir o ambiente em que se inserem e providenciam os dados necessários das variáveis que monitorizam. Nesta dissertação irão ser utilizados diversos sensores, com os protocolos de comunicação SPI 3.2.3.1e o I_2C 3.2.3.2. Os sensores que poderão ser adicionados à arquitetura, ainda por implementar, são os de temperatura, humidade do ar, humidade do solo, pressão e ainda sensores de gases que possibilitam a avaliação da qualidade do ar.

3.4.1 Max38155 com o TMP36

Um dos sensores de temperatura a usar é composto por duas componentes, o termopar TMP36 e o conversor A/D Max38155.

O termopar TMP36, é do tipo k e permite sentir temperaturas desde os -200 [°C] até aos 1350 [°C], consulte os sites [53].

A segunda componente é o conversor Analógico/Digital Max38155, que além de permitir a amplificação do sinal de entrada, também converte os valores de tensão analógicos do termopar TMP36, para valores de tensão digitais usando como recurso o protocolo de comunicação SPI. Informação exposta no site [52].

Estes dois módulos têm aplicações em sistemas HVAC, Industriais, como são exemplos a industria automóvel e a produção eletrodomésticos.

3.4.1.1 Especificações Técnicas

A Tabela 3.7, indica as especificações do termopar TMP36 do tipo k e a Tabela 3.8 apresenta as especificações do conversor A/D Max38155.

Tabela 3.7: Especificações do termopar TMP36 tipo K

		Unidade
Resolução	50	[uV/°C]
Intervalo de Medição	-100 a 500	[°C]
Precisão	± 2	[°C]
Tensão de Saída V_{out}	-6 a 20	[mV]

Tabela 3.8: Especificações do conversor Analógico/Digital MAX38155

Intervalo de Medição [°C]	-200 a 1350
Precisão [°C]	$\pm 2at \pm 6$
Alimentação V_{CC} **[V]**	3.3 ou 5
Interface de Comunicação	"SPI"
Compatibilidade com o termopares	tipo k

A Tabela 3.9 apresenta a descrição dos terminais do conversor A/D Max38155.

Tabela 3.9: Descrição e Função dos Terminais do componente MAX38155

Terminal	Descrição
Vin	Alimentação a 5[V]
3vo	Alimentação a 3.3[V]
GND	Ground
DO	Data Output
CS	Chip Select
CLK	Clock

3.4.2 DHT11

O sensor DHT11 é simulataneamente um sensor de temperatura e humidade. Ao usar módulos com tecnologia digital assegura uma maior precisão nas leituras e maior durabilidade e estabilidade do equipamento. Este sensor é composto por uma resistência sensível à humidade e um termístor NTC, que é sensível à temperatura e são ligados a um micro-controlador, que opera a 8 bit conforme o datasheet [1].

O sensor DHT11 normalmente é utilizado em estações meteorológicas, equipamentos médicos, indústria automóvel, sistemas de ar condicionados e desumidificadores, eletrodomésticos, inspeções e testes a equipamentos diversos, controlo automático reguladores de humidade.

3.4.2.1 Especificações Técnicas

A Tabela 3.10 representa as especificações técnicas do sensor DHT11 e a Tabela 3.11 apresenta as suas características elétricas.

Tabela 3.10: Especificações do sensor DTH11

	Humidade Relativa	Temperatura
Resolução	16 bit	16 bit
Intervalo de Medição	20 a 90%[RH]	0 a 50 [°C]
Precisão a 25 [°C]	± 5 %[RH]	± 2 [°C]
Repetibilidade	$\pm 1\%[RH]$	$\pm 0.2[C]$
Tempo de resposta	1/e (63%) a 25 [°C] 6S	1/e (63%) 10S
Histerese	$< \pm 0.3\%[RH]$	-
Estabilidade a longo prazo	$< \pm 0.5[RH]/ano$	-

A utilização deste sensor requer cuidados especiais na sua utilização. O local e a montagem do mesmo deverá ter em consideração os seguintes aspetos, caso contrário o seu desempenho poderá ser afetado:

- Vapores derivados de substâncias químicas;

Tabela 3.11: Características elétricas do sensor DHT11

Alimentação	3.3 a 5 [V] DC
Corrente	Durante as medições 0.3 [mA] e em repouso 60 [µA]
Período de Amostragem	+ 2 [s]

Tabela 3.12: Terminais DTH11

Terminal	Descrição
1	Alimentação de 3.3 a 5 [V] DC
2	Serial Data, um único bus de comunicação
3	-
4	Ground

- A longa exposição a fontes de luz forte e a luz ultravioleta;
- A distância entre a placa de aquisição de dados e o sensor deverá ser a mínima possível;
- Durante o processo de soldadura deverá garantir que a temperatura desta não ultrapasse os 260 [°C];
- Há que ter presente que a leitura da humidade depende da temperatura.

A não verificação destes requisitos poderá requerer a frequente calibração deste tipo de sensores.

3.4.2.2 Calibração

A calibração permite que o equipamento em questão forneça dados corretos relativamente às variáveis que este monitoriza. O processo de calibração do sensor DHT11 faz-se da seguinte forma:

- Manter o sensor DTH11 a uma temperatura entre 50 a 60 [°C] e a humidade <10 [%RH] durante 2 horas;
- Manter o sensor DTH11 a uma temperatura entre 20 a 30 [°C] e a humidade >70[%RH] durante 5 horas.

3.4.3 MPL115A2

O barómetro e termómetro MPL115A2 é um sensor digital com interface I_2C, tem baixo consumo energético, e devido à sua dimensão compacta é utilizado em projetos em que o espaço para eletrónica é reduzido. Os dados de calibração por defeito estão guardados

numa ROM interna, o que evita deslocá-lo um laboratório externo certificado para a respetiva calibração, ver informação no datasheet [32].

As aplicações para este sensor são barómetros, altímetros, estações meteorológicas, discos rígidos (HDD), equipamento industrial, monitorização na saúde e sistemas de controlo de aviação.

3.4.3.1 Especificações Técnicas

As especificações técnicas do sensor MPL115A2 são apresentados na Tabela 3.13.

Tabela 3.13: Especificações do sensor "MPL115A2"

	Temperatura	**Pressão**
Resolução	-	$0.15[kPa]$
Intervalo	-40 [°C] a $+105$ [°C]	50 [kPa] a 115 [kPa]
Precisão	-	± 1[kPa]
Tempo de Conversão	-	3 [ms]
Tempo de Arranque	-	5 [ms]

Na Tabela 3.13, é mencionado o Tempo de Conversão e o Tempo de Arranque. Estes dependem do início da contagem até ao seu términus, o que significa que estes dois conceitos são únicos para este tipo de sensor.

- **Tempo de Conversão**: Tempo decorrido entre o início do comando de conversão e a disponibilização dos dados de Temperatura e de Pressão.
- **Tempo de Arranque**: Tempo decorrido entre a saída do modo desligar até à comunicação com o dispositivo para iniciar o comando de leitura dos dados.

As características elétricas do sensor MPL115A2 são apresentados na Tabela 3.14 e os seus terminais e descrição são apresentados na Tabela 3.15

Tabela 3.14: Características Elétricas "MPL115A2"

Alimentação	$2.375a5.5[V]$
Corrente em modo ativo	$5[\mu A]$
Corrente em mod sleep	$1[\mu A0]$

3.4.4 MQ2

O componente MQ2 é um sensor com a capacidade de detetar vários gases, entre eles o Hidrogénio (H_2), Gás de petróleo liquefeito (GPL), Metano (CH_4), Álcool e Propano. A sua rapidez de processamento permite detetar a presença de um dado gás de forma rápida.

Tabela 3.15: Descrição dos terminais do sensor MPL115A2

Terminal do Sensor MPL115A2	Descrição
VDD	2.375a5.5 [V]
GND	Massa ou Terra
SDA	"*Send Data*"
SCL	"*Serial Clock*"

Este sensor é constituído por uma bobina dentro de uma câmara, onde o gás está presente e a variação da corrente na bobina identifica qual o gás em presença. Para além da bobina existe um potenciómetro que permite ajustar a sensibilidade do sensor MQ2. Consulta o datasheet [34].

Podemos encontrar o sensor MQ2 em sistemas de deteção de fugas de gás em espaços domésticos e em ambiente industrias.

3.4.4.1 Especificações Técnicas

A Tabela 3.16 especifica as características elétricas do sensor MQ2.

Tabela 3.16: Características Elétricas MQ2

Itens	Descrição	Min	Tipo	Max	Unidade
V_{CC}	Alimentação	4.9	5	5.1	[V]
V_H	Tensão de Calor	4.9	5	5.1	[V]
P_H	Potência dissipada pelo calor	0.5	-	800	[mW]
R_L	Resistência de carga	-	Ajustável	-	[Ω]
R_H	Resistência de calor	-	33	-	[Ω]
R_S	Resistência do sensor	3	-	30	[kΩ]

3.4.4.2 Circuito de teste do sensor MQ2

O sensor MQ2 foi submetido a um teste para calcular algumas especificações, em conjunto com um circuito básico de integração. O teste consistiu em inserir o sensor MQ2 num circuito de teste numa sala especial, com a capacidade de injetar um gás selecionado e de controlar a sua quantidade presente no espaço, visite o datasheet [34].

A Figura 3.3, adaptada do datasheet [34] presente na pág. 1 , é um circuito para testar o funcionamento do sensor MQ2. A tensão de aquecimento (V_H) necessita de pelo menos 2 [V] á entrada, pois o sensor precisa de aquecer para que as suas leituras tenham a maior precisão possível. Esta tensão pode ser maior que 2 [V], isto não significa que ao subir linearmente a tensão de aquecimento (V_H) melhore a sua performance. Neste caso a tensão de aquecimento é igual à tensão de alimentação (V_{CC}).

$$V_H = V_{CC} \tag{3.1}$$

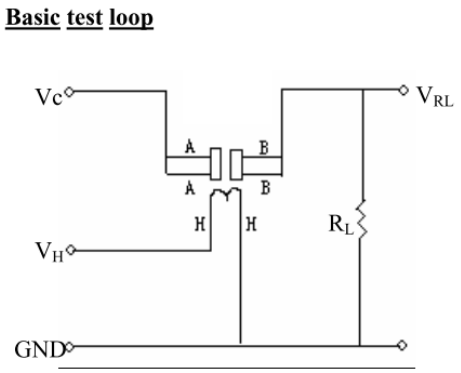

Figura 3.3: Circuito de teste do sensor "MQ2"

A tensão de alimentação (V_{CC}) permite que haja uma tensão de saída em relação à carga (R_L), ou seja permite calcular a resistência de carga (R_L) que por sua vez está em paralelo com o sensor.

A potência de sensibilidade pode ser calculada da seguinte forma:

$$P_S = V_{CC}^2 \frac{R_S}{R_S + R_L} \tag{3.2}$$

A resistência do gás pode ser calculada pela seguinte fórmula:

$$R_S = R_L[\frac{V_{CC}}{V_{RL}} - 1] \tag{3.3}$$

A Equação 3.3 V_{CC} é a tensão de alimentação e o V_{RL} é a tensão na resistência de carga.

A Tabela 3.17 expõe as especificações técnicas do sensor MQ2, em conjunto com o circuito de teste indicado na Figura 3.3 nas condições de teste representadas na Tabela 3.17.

O sensor MQ2 foi aquecido durante 48 horas antes de se iniciar o processo de medição.

A Figura 3.4,adaptada do datasheet [34] presente na pág. 2, poderemos observar um gráfico que apresenta a variação do ganho $\frac{R_S}{R_O}$ ao longo da variação da concentração de um certo gás em [ppm].

A Figura 3.5,adaptada do datasheet [34] presente na pág.2, podemos observar qual o ganho $\frac{R_S}{R_O}$ ao longo da temperatura [°C], em várias Humidades Relativas constantes em [% RH].

Tabela 3.17: Condições de teste ao sensor MQ2 em conjunto com o circuito de teste

Itens	Descrição	Min	Tipo	Max	Unidade
V_{CC}	Alimentação	4.9	5	5.1	[V]
V_H	Tensão de Calor	4.9	5	5.1	[V]
RH	Humidade Relativa	60	65	70	[% RH]
T	Temperatura	18	20	22	[°C]

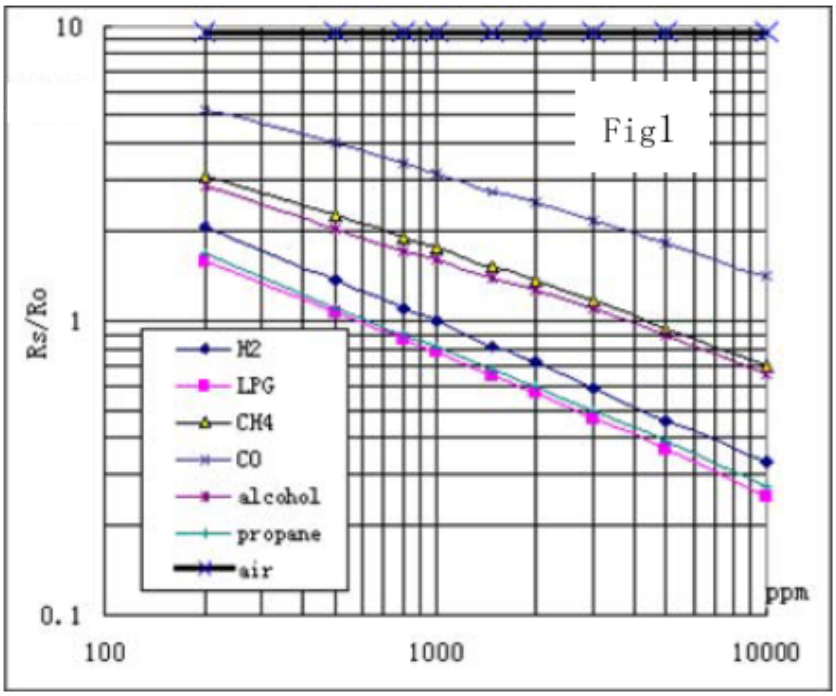

Figura 3.4: Gráfico Rs/Ro por ppm

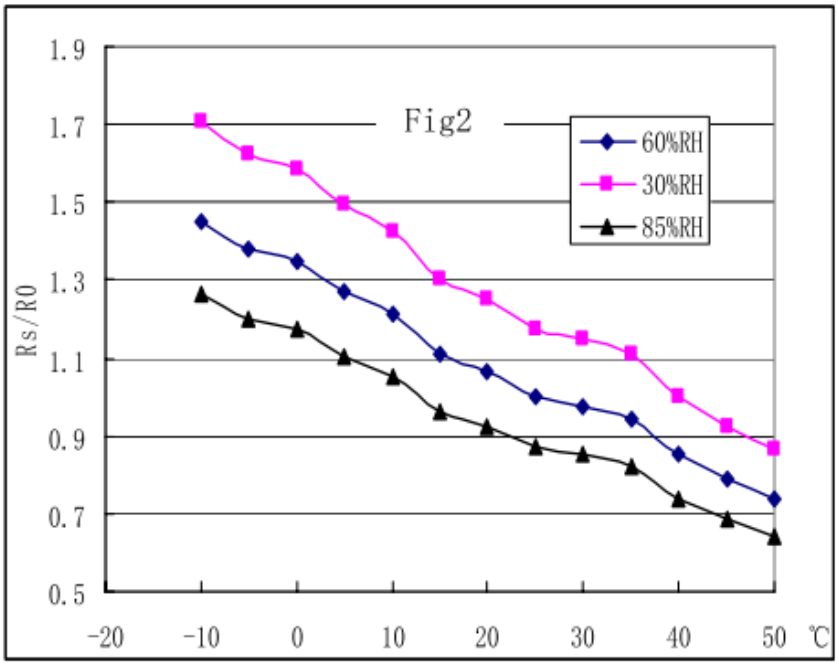

Figura 3.5: Gráfico Rs/Ro por ºC

Tabela 3.18: Especificações Técnicas do sensor MQ2 em conjunto com o circuito de teste

Itens	Descrição	Min	Tipo	Max	Unidade
V_{CC}	Alimentação	-	≤ 24	-	[V] DC
V_H	Tensão de Calor	4.8	5	5.2	[V]
P_H	Potência dissipada pelo calor	-	≤ 900	-	[mW]
R_L	Resistência de carga	-	Ajustável	-	[Ω]
R_H	Resistência de aquecimento	29	31	34	[Ω]
R_S	Resistência do sensor	-	2 a 20	-	[kΩ]
S	Sensibilidade	-	$\frac{R_S(ar)}{Rs} \geq 5$	-	-
ff	Declive	-	$\leq 0.6 \frac{R_{5000ppm}}{R_{3000ppm}}$	-	-

Na Tabela 3.18 é importante referir que o valor de resistência de carga (R_S) foi calculado quando a sala continha 1000 [ppm] do gás C_3H_8. A sensibilidade (S) pode ser calculada usando pela Equação 3.4.

$$S = \frac{R_{S_ar}}{Rs} \tag{3.4}$$

Mas no entanto esta fórmula foi determinada na condição em que o gás que presente na sala era o CH_4 com 1000 [ppm], logo o declive indicado é o verificado para este caso.

3.4.5 MG811

O sensor MG811 permite detetar os gases de Dióxido de Carbono (CO_2), Etanol($C_2H_5 - OH$), Monóxido de Carbono (CO) e o Metano (CH_4). No entanto só é capaz de quantificar o gás CO_2 presente em partes por milhão. O funcionamento deste sensor é muito idêntico ao sensor MQ2, vide a secção 3.4.4 no §2.

Este sensor é constituído por uma câmara com uma bobina. A corrente da bobina é afetada pela presença do gás na câmara. Por outro lado este sensor é pouco afetado pela temperatura e a humidade presente na câmara, o que possibilita a determinação da temperatura e humidade do ambiente. Para mais informações consulte o datasheet [30]. Este sensor pode ser encontrado em sistemas de controlo de qualidade do ar, controlo de processos de fermentação e cálculo de temperatura através da concentração de CO_2.

3.4.5.1 Especificações Técnicas

A Tabela 3.19 apresenta as especificações técnicas do sensor MG811.

A Tabela 3.20 relata os terminais deste sensor e sua descrição.

A Figura 3.6 ,adaptada de [30] presente na pág. 2, expõe um gráfico de variação da concentração do gás dióxido de carbono versus tensão "EMF"em [mV]. Também podemos observar que o sensor MG811 também pode detetar outros gases, tais como, Etanol($C_2H_5 - OH$), Monóxido de Carbono (CO) e o Metano (CH_4). No entanto estes gazes como apresentam valores de tensão muito próximo, torna-se difícil sua distinção.

Tabela 3.19: Especificações Técnicas do sensor MG811

Itens	Descrição	Min	Tipo	Max	Unidade
V_{CC}	Tensão de Alimentação DC	-	5	-	[V]
V_H	Tensão de Calor	5.9	6	6.1	[V]
P_H	Potência dissipada pelo calor	-	1200	-	[mW]
I_H	Corrente de Aquecimento	-	200	-	[mA]
R_H	Resistência de aquecimento	25	30	35	[Ω]
Tao	Temperatura de operação	20	- a 70	-	[°C]
Tas	Temperatura de Armazenamento	30	-	50	[°C]

Tabela 3.20: Descrição dos terminais do sensor MG811

Terminal do Sensor MG811	Descrição
V_{CC}	Alimentação 5 [V] em DC
GND	Massa ou Terra
A0	Porta analógica

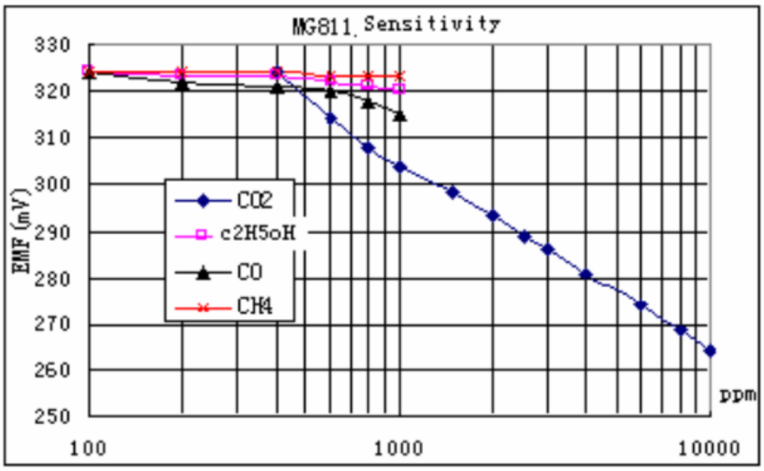

Figura 3.6: Gráfico EMF [mV]/ppm

Porém o Monóxido de Carbono a partir dos 600 [ppm] distancia-se dos restantes gases tendo diferenças significativas de 10 [mV].

3.4.6 Humidade do Solo

O sensor de Humidade do Solo permite detetar e determinar o valor de humidade presente no solo. Este é constituído por duas chapas metálicas e um potenciómetro. As chapas metálicas, quando colocadas no solo, criam uma diferença de potencial. A tensão varia pelas duas chapas metálicas dependendo do meio em que são inseridos e da quantidade de água presente. O potenciómetro serve apenas para ajustar um valor de tensão "*threshold*". Quando as chapas metálicas detetarem um valor de tensão acima do valor "*threshold*", o sensor envia um sinal digital de 5 [V] para a porta de digital. Infrome-se no

datasheet [13].

3.4.6.1 Especificações Técnicas

A Tabela 3.21 apresenta as especificações elétricas do sensor Humidade do Solo.

Tabela 3.21: Características Elétricas Humidade do Solo

Variável	Descrição	Min	Tipo	Max	Unidade
V_{CC}	Tensão Alimentação DC	3.3	-	5	[V]
I	Corrente	0	-	5	[mA]
T_{AO}	Tensão de saída na porta analógica	0	-	5	[V]
T_{DO}	Tensão de saída na porta digital	0	-	5	[V]
OVD	Constante de saída em solo seco	0	-	300	-
OVH	Constante de saída em solo húmido	300	-	700	-
OVW	Constante de saída em água	700	-	950	-
Histerese	-	-	$V_{CC} * 0.09$	-	[V]

A Tabela 3.22 exibe os terminais deste sensor e a sua função.

Tabela 3.22: Descrição dos terminais do sensor Humidade do Solo

Terminal do Sensor Humidade do Solo	Descrição
V_{CC}	Alimentação 5 [V] em DC
GND	Massa ou Terra
A0	Porta analógica

3.5 Dispositivos Periféricos

3.5.1 Leitor de Cartões micro SD

O modulo "SKU:DFR0229"é um leitor de cartões microSD compatível com TF SD Card. O protocolo de comunicação com o micro-controlador é o SPI 3.2.3.1, opera a uma tensão de alimentação em DC de 5 [V]. As suas dimensões são 20x28 [mm].

Este módulo pode ser encontrado em sistemas de áudio, vídeo, gráficos e em expansões de memória para dispositivos com memória volátil.

3.5.1.1 Interface com o Micro-Controlador "Arduino"

A Tabela 3.23, apresenta os terminais do módulo de leitor de cartões microSD com a sua descrição e a que terminal do mico-controlador "*Arduino*"se deve conectar. Para informação mais detalhada consulte [31].

Para além de ser necessário conectar os terminais de forma correta, é necessário criar a programação que realiza a gestão do módulo de leitor de cartões micro SD. Para o fazer

Tabela 3.23: Terminais do módulo leitor de cartões microSD

Terminal	Descrição	micro-controlador "Arduino Uno"
MISO	"*Master Input Slave Output*"	12
SCK	"*Serial Clock*"	13
SS	"*Slave Select*"	4
MOSI	"*Master Output Slave Input*"	11
GND	"*Ground*"	GND
V_{CC}	"*Tensão de alimentação*"	5 [V]

é necessário a utilização da biblioteca "SD.h", que vem já incluída na aplicação "*Arduino IDE*". Informação exposta no capítulo "Anexos"na secção "Bibliotecas"I.2.0.1 no parágrafo 2.

3.5.2 Real Time Clock

Um módulo "*Real Time Clock*"é um circuito integrado que possibilita a gestão das funções do tempo. Normalmente este tipo de módulo conta o tempo em segundos, minutos, horas, dias, meses e anos. Na grande maioria dos casos, estes módulos contêm circuitos para a gestão de uma bateria externa, o que permite que o circuito principal não pare a contagem do tempo, durante a vida útil da bateria. Informação expostas nos datasheets [26] e [27].

3.5.2.1 DS1307 com EEPROM

O módulo é composto por dois circuitos integrados, o DS1307 e a EEPROM AT24C32. A este módulo também pode ser ligado um sensor de temperatura externo DS18B20, que para garantir a independência energética contém uma interface para uma bateria.

O circuito integrado DS1307 é um "*Real Time Clock*"que gere as funções do tempo. Estas funções são a disponibilização da informação relativa aos segundos, minutos, horas, dias, meses e anos, opera ainda com os formatos de hora 24 horas ou 12 horas AM/PM e gere automaticamente meses com menos de 31 dias. O circuito integrado DS1307 contém um circuito que permite detetar a perda de alimentação externa e como consequência ativa a alimentação da bateria. Este módulo contém ainda uma RAM de memória não volátil de 56 Byte e comunica via I_2C 3.2.3.2, consome menos de 500 [nA], tem uma tensão de alimentação em DC de 5 ± 0.5 [V] e funciona numa gama de temperaturas de -40 a 85 [°C].

A EEPROM AT24C32, comunica através do protocolo I2C, tem 32 [kB] de memória e permite o armazenamento de palavras de 8 bit, opera a uma tensão de 1.8 a 5.5 [V] a corrente a 5.5 [V] é de 2 [μ A].

O módulo é composto por 13 terminais, dos quais 7 são para o acesso ao RTC DS1307 e os restantes são para o acesso à EEPROM. A Tabela 3.24, indica os terminais, a sua função e a que circuito integrado pertencem. Consulte o datasheet [27].

Tabela 3.24: Terminais do módulo RTC DS1307 com EEPROM AT24C32

Terminal	Descrição	Dispositivo
BAT	Terminal de Tensão da Bateria	RTC
V_{CC1}	Terminal de alimentação	RTC
V_{CC2}	Terminal de alimentação	EEPROM
GND	"*GND*"	RTC
GND	"*GND*"	EEPROM
SDA	"*Serial Data*"	RTC
SDA	"*Serial Data*"	EEPROM
SCL	"*Serial Clock*"	RTC
SCL	"*Serial Clock*"	EEPROM
SO	""	RTC

3.6 Placas de Comunicação

3.6.1 nRF24l01+

A placa de comunicação nRF24L01+, fabricado pela "*Nordic SemiConductors*", possibilita a criação de uma rede sem fios "*Mesh*", informação exposta no capítulo "Anexos"na secção "Comunicações: Mesh"I.3.1. Esta placa opera na banda ISM a 2.4 [GHz] e permite ter velocidades de transmissão a 250 [Kbps], 1 [Mbps] e 2 [Mbps] com 126 canais RF. Este dispositivo tem baixo consumo energético em que a corrente no canal de transmissão é de 11.3 [mA] e no canal de receção é de 13.5 [mA] e a tensão de alimentação em DC está compreendida entre 1.9 a 3.6 [V].

Durante as comunicações, este dispositivo gere automaticamente os pacotes de receção e de transmissão. Este componente é compatível com as placas nRF2401A, nRF2402, nRF24E1 e nRF24E2, o que possibilitou que estas estejam todas na mesma rede "*Mesh*". O componente nRF24L01+ aplica-se como um periférico sem fios de um computador, proporcinando uma ligação de dispositivos periféricos tais como um rato e um teclado. A placa pode ser encontrada em centros de media, relógios de desporto, redes de sensores de baixo consumo e controladores remotos de RF. Informe-se no datasheet [44]. As Tabelas: 3.25, 3.26 e 3.27 apresentam a descrição das características elétricas, as características elétricas e funções de cada terminal. Informe-se com o datasheet [44]

Tabela 3.25: Descrição das características elétricas do módulo de comunicação nRF24L01+

Variável	Descrição
V_{DD}	Tensão de alimentação em DC
P_D	Potência dissipada
I_{DD_PD}	Corrente de alimentação a desligar
I_{VDD_ST1}	Corrente de alimentação no modo I
I_{VDD_ST2}	Corrente de alimentação no modo II
I_{VDD_TX0}	Corrente de alimentação com o terminal TX a 0 dBm
I_{VDD_TX6}	Corrente de alimentação com o terminal TX a -6 dBm
I_{VDD_TX12}	Corrente de alimentação com o terminal TX a -12 dBm
I_{VDD_TX18}	Corrente de alimentação com o terminal TX a -18 dBm
I_{VDD_TXS}	Corrente de alimentação durante o estabelecimento do terminal TX
I_{VDD_2M}	Corrente de alimentação a 2 [Mbps]
I_{VDD_1M}	Corrente de alimentação a 1 [Mbps]
I_{VDD_250}	Corrente de alimentação a 250 [kbps]
I_{VDD_RXS}	Corrente de alimentação durante o estabelecimento do terminal RX
TO	Temperatura de operação
TA	Temperatura de armazenamento

Tabela 3.26: Características da placa de comunicação nRF24L01+

Variável	Min	Tipo	Max	Unidade
V_{DD}	1.9	3.0	3.6	[V]
P_D	-	60	-	[mW]
I_{DD_PD}	-	900	-	[nA]
I_{VDD_ST1}	-	26	-	[μ A]
I_{VDD_ST2}	-	320	-	[μ A]
I_{VDD_TX0}	-	11.3	-	[mA]
I_{VDD_TX6}	-	9.0	-	[mA]
I_{VDD_TX12}	-	7.5	-	[mA]
I_{VDD_TX18}	-	7.0	-	[mA]
I_{VDD_TXS}	-	8.0	-	[mA]
I_{VDD_2M}	-	13.5	-	[mA]
I_{VDD_1M}	-	13.1	-	[mA]
I_{VDD_250}	-	12.6	-	[mA]
I_{VDD_RXS}	-	8.9	-	[mA]
TO	-40	+27	+85	[°C]
TA	-40	-	+125	[°C]

Tabela 3.27: Descrição dos Terminais da placa de comunicação componente NRF24L01

Terminal	Nome	Descrição
1	GND	Terminal de massa comum
2	VCC	Terminal de Alimentação
3	CE	"*Chip Enable*"
4	CSN	"*SPI Chip Select*"
5	SCK	"*SPI Clock*"
6	MOSI	"*SPI Slave Data Input*"
7	MISO	"*SPI Slave Data Output*"
8	IRQ	"*Maskable Interrupt (Active Low)*"

3.6.1.1 Interface com o micro-controlador Arduino

A placa de comunicação nRF24L01+ pode ser integrada com o micro-controlador "Arduino Uno"ou "Nano". Para fazer essa integração por hardware é preciso conectar os terminais do componente nRF24L01+ a uns terminais específicos do micro-controlador "Arduino Uno"ou "Nano". A Tabela 3.28 apresenta o modo como se deve conectar est módulo ao micro-controlador.

Tabela 3.28: Integração do componente nRF24L01+ com o micor-controlador Arduino

Terminal nRF24L01+	Terminal Micro-Controlador Arduino
1- GND	GND
2- VCC	3.3V
3- CE	9
4- CSN	10
5- SCK	13
6- MOSI	11
7- MISO	12
8- IRQ	-

A biblioteca que permite a integração do rádio nRF24L01+ aos micro-controladores "Arduino"é "RF24NETWORK.h", que contém as funções necessárias para a gestão das mensagens trocadas entre micro-controladores. Com a biblioteca estão disponíveis dois programas para a comunicação entre dois micro-controladores "Arduino"com o componente nRF24L01+. Os programas são "*hellowrold_rx.ino*"e o "*helloworld_tx.ino*". As funções gerem das comunicações entre micro-controladores com a placa de comunicação nRF24L01+ são:

- RF24 Radio(MOSI, MISO) - Construtor que permite a definição dos terminais MOSI e MISO por software;
- RF24Network network(radio); - Construtor para a biblioteca RF24NETWORK.h que permite a definição do radio;

- SPI.begin() - Inicialização do SPI;
- radio.begin() - Inicialização do radio;
- network.begin(/ ∗ *channel* ∗ /90, / ∗ *nodeaddress* ∗ / $this_node$) - Criação da rede "*Mesh*", o primeiro parâmetro é o canal, o segundo parâmetro é o endereço do "*Node* de destino e o último parâmetro é o endereço do "*Local*";
- network.update() - Função que realiza a sincronização das novas mensagens;
- network.available() - Função que verifica a receção de novas mensagens;
- network.read(*header*,&*payload*,*sizeof*(*payload*)) - Função que copia a informação do buffer para uma estrutura de dados;
- network.write(*header*,&*payload*,*sizeof*(*payload*)) - Função que permite a escrita da mensagem para um buffer, para esta ser enviada. Esta função também verifica se a mensagem chegou ao destino e devolve num variável booleana "True"em caso de sucesso e "False"em caso de falha.

A informação das mensagens fica acessível na estrutura de dados:

Listagem 3.1: Estrutura de dados das mensagens

```
1 struct payload_t
2 {
3   unsigned long ms;
4   unsigned long counter;
5 };
```

A biblioteca "RF24NETWORK.h"é dependente de outras duas bibliotecas "RF24.h"e a "SPI.h".

3.6.2 BT-HC06

A placa de comunicação BT HC-06 cria uma rede sem fios "*Bluetooth V2.0+EDR (Enhanced Data Rate)*". Este módulo, dependendo da modulação, permite uma velocidade de transmissão de dados de 3 [Mbps] e opera na banda 2.4 [GHz]. Este tem um modo de baixa potência, com uma tensão de alimentação a 1.8 [V], no entanto permite valores de tensão compreendidos entre 1.8 a 3.6 [V]. A potência de transmissão é de +4 [dBm] e contêm uma antena integrada. Este módulo possibilita a interligação com um micro-controlador externo usando o protocolo de comunicação "UART"3.2.3.4. O módulo BT HC-06 tem a capacidade de se conectar novamente ao último dispositivo "*Bluetooth*"conhecido, em caso de falha, possibilita ainda a definição de um dispositivo para emparelhamento por defeito. A password de acesso a este dispositivo é "0000"ou "1234". Informação exposta no datasheet [14]. A Tabela 3.29 indica as especificações técnicas, a Tabela 3.30 apresenta a configuração do protocolo de comunicação "UART"e por fim a TabelaI.5 mostrará os terminais do módulo BT HC-06 e suas funções.

Tabela 3.29: Especificação Técnica do módulo BT-HC06

Descrição	BT HC-06
Protocolo de Bluetooth	BT 2.0+EDR
Frequência [GHz]	2.4
Modulação	GFSK
Potência de Emissão [dBm]	Classe 2
Potência de Transmissão[dBm]	<=-84 a 0.1% Erro
Velocidade Assincrona [Mbps]	2.1
Velocidade Sincrona [Kbps]	160
Tensão de entrada [V]	3.3
Corrente de entrada [mA]	50
Temperatura em funcionamento[°C]	-20 até 55
Master	Não
Slave	Sim
Comandos AT	Sim

Tabela 3.30: Configuração do protocolo de comunicação UART para o módulo BT-HC06

"*Data bits*"	8
"*Stop bit*"	1
Paridade	Não
Velocidades de Transmissão	9600, 19200, 38400, 57600, 115200, 230400, 460800

Tabela 3.31: Terminais do módulo BT-HC06

Terminal	Nome	Função
1	V_{CC}	Terminal de massa comum
2	GND	Terminal de alimentação
3	TXD	Terminal de transmissão
4	RXD	Terminal de receção

3.6.2.1 Interface com o micro-controlador Arduino

Visto que a placa de comunicação BT HC-06 cria uma ligação a um micro-controlador externo via protocolo de comunicação "UART", em primeiro lugar, é necessário garantir que os terminais estejam ligados corretamente. A Tabela 3.32 indica o modo.

A programação gere as comunicações entre a placa de comunicação BT HC-06 e o micro-controlador "Arduino"segue o standard de comunicação série do "*Arduino*", consulte na secção "Programação: Comunicação Serie"I.2.1.2 no capítulo "Anexos".

Tabela 3.32: Ligação entre os terminais do módulo BT HC-06 e o Arduino

Terminal BT HC-06	Micro-Controlador Arduino
TXD	RX
RXD	TX
V_{CC}	3.3V
GND	GND

3.7 Raspberry Pi

O mini-pc "Raspberry Pi"é um computador de baixo custo, de pequenas dimensões, com interface para se conectar a um ecrã, teclado, rato e até à Internet. O mini-pc foi desenvolvido no Reino Unido pela Fundação "Raspberry Pi"em que o seu principal objetivo é promover a educação nas áreas da ciência, da tecnologia e lógica computacional em escolas, e em universidades. A Fundação Raspberry Pi desenvolveu vários modelos do mini-pc "Raspberry Pi", tais como, o modelo A, B, B+, 2, Zero, e o 3. Consulte o site [41].

3.7.1 Raspberry Pi Modelo B

As Tabela 3.33 e 3.34 apresentam as especificações técnicas do mini-pc "Raspberry Pi B", conforme os sites [39] e [42].

Tabela 3.33: Especificação do mini-pc "Raspberry Pi B"1

Itens	Especificação	Unidades
CPU	ARM11 ARM1176JZF-S a 700	[MHz]
GPU	Broadcom VideoCore IV a 250	[MHz]
RAM	512	[MB]
Portas USB (2.0)	2	unidade
Porta Full HDMI	Sim	-
Video - RCA	Sim	-
Entrada 3.5 [mm] de audio	Sim	-
Interface para Ecrã	Sim	-
Armazenamento de dados	SD Card	-
Terminais GPIO	26	unidade
UART	Sim	-
I_2C	Sim	-
Terminais IDC	Não	
SPI	Sim	-
Porta de Ethernet	Sim	-
Wi-Fi	Não	-
Bluetooth	Não	-

Tabela 3.34: Especificação do mini-pc "Raspberry Pi B"2

Itens	Especificação	Unidades
OpenGPL ES	2.0	-
OpenVG	180p30 H.264 high-profile encode/decode	-
Alimentação V_{CC}	5	[V]
Corrente	700	[mA]
Potência	3.5	[W]
Comprimento]	85.0	[mm]
Largura	56.0	[mm]
Altura	17	[mm]
Peso	40	[g]

3.7.2 Sistemas Operativos para o mini-pc Raspberry Pi

Um Sistema Operativo é um programa ou um conjunto de programas que gerem os recursos do sistema, fornecendo uma interface a um utilizador, para este possa operar com o sistema. Os sistemas operativos para os mini-pc "*Raspberry Pi*"na sua grande maioria são baseados em sistemas "*Linux*". Para esta dissertação foram considerados dois sistemas operativos: - o sistema operativo "Noobs", - o sistema operativo "Raspbian".

1. O Sistema Operativo "Noobs"é baseado no "*Linux Debian* foi simplificado para que utilizadores básicos de sistemas linux possam operar com este sistema operativo. Este sistema está bastante simplificado e está moldado para a performance da máquina.

2. O sistema operativo "Raspbian"já é um sistema mais completo.

Consulte os sites [37] e [17].

3.7.3 Firewall

Uma "*Firewall*"é um sistema de segurança para uma rede, é constituído por protocolos que contém funções de segurança. Existem diversos dispositivos que possibilitam a criação de redes, tais como, "access points", "routers", "switches"e "repeaters"que contém "firewalls"embutidas nos seus sistemas, consulte o sitio [35].

3.7.4 Protocolo *Secure Shell* (SSH)

O protocolo encriptado Secure Shell - SSH opera serviços, de uma forma segura numa rede. O protocolo SSH opera dentro de um canal seguro numa rede entre o cliente e o servidor, informe-se no site [49].

3.8 Virtual Private Network

Uma "Virtual Private Network", mais conhecida por VPN é uma extensão da rede de uma rede privada numa rede pública. Este tipo de rede envia e recebe a informação através de uma rede partilhada ou uma rede pública. A "VPN"aumenta a segurança encriptando as comunicações e usa o protocolo "Tunneling". Informação exposta no site [55].

3.8.1 Aplicações

Existem inúmeras aplicações para uma "Virtual Private Network - VPN"algumas aplicações para o seu uso:

- Acesso a redes corporativas enquanto o seu utilizador está fora do escritório;
- Interoperabilidade em escritórios de localização geográfica diferente;
- Aumentar a segurança;
- Acesso a servidores e serviços locais.

O protocolo "Tunneling"é um protocolo que comunica através de um canal especifico do protocolo TCP que se chama GRE. No canal GRE, a VPN encpsula os pacotes PPP.

3.8.2 Point-to-Point Tunneling

A "virutal Private NetworkPoint-to-Point Tunneling"comunica através de um canal específico do TCP, chamado GRE. Este canal envia os pacotes encriptados PPP. A comunicação é feita usando o TCP na porta 1723. Esta comunicação é usada para iniciar e gerir o canal GRE, informe-se [43].

3.9 Apache

O projeto "Apache"tem como objetivo desenvolver e gerir um servidor HTTP para os sistemas operativos mais modernos, como sistemas "unix"e "windows". Um dos grandes objetivos é providenciar serviço HTTP mais seguro, escalável é eficiente. Os serviços HTTP regem-se pelos standard correntes do HTTP, visite o site [51].

CAPÍTULO 4

IMPLEMENTAÇÃO

Na fase embrionária da implementação deste projeto estava previsto utilizar o micro-controlador "*Pinoccio*"para criar uma rede de sensores. Este micro-controlador é constituído por um processador com um rádio RF integrado, que opera a 2.4 GHz. É disponibilizado pela empresa "*Pinoccio*", uma extensão para o browser "*Google Chrome*", uma interface capaz de integrar vários micro-controladores "*Pinoccio*"na plataforma. Esta plataforma é uma interface programável para a visualização de dados sentidos pelos possíveis sensores que pudessem ser integrados com este micro-controladores. Estes dados eram comunicados utilizando a tecnologia "*Wi-Fi*".

Existiram vários problemas que implicaram colocar de parte esta solução para o problema e criar uma nova solução.

Os problemas que existiam eram de natureza de hardware e de software. No lado do hardware existiam problemas de alimentação intermitente, as comunicações por vezes não eram estabelecidas. Havia ainda dificuldades na criação de redes pois a tecnologia não o permitia. No lado do software o carregamento dos programas desenvolvidos para o micro-controlador não era possível efetuar e na plataforma não era possível realizar a integração dos vários micro-controladores, impossibilitando a visualização dos dados.

Estes obstáculos levaram a propor ao professor orientador da dissertação criar um novo sistema de sensores utilizando outras tecnologias.

A solução encontrada foi usar vários micro-controladores "Arduino"com o módulo de comunicação "*Mesh*", em que cada micro-controlador seria um "*Hop*"com vários sensores integrados.

Como é necessário garantir o registo de dados sentidos pelos sensores e visto que os micro-controladores não têm uma ligação à Internet, foi necessário estender a arquitetura. A extensão passou por integrar um "*Smartphone*"que recebia os dados sentidos pelos sensores e enviá-los para um servidor com uma base de dados. Os dados obtidos pelos

sensores poderiam ser visualizados num site desenvolvido para esse efeitoS.

Como consequência a implementação desta arquitetura foi dividida em três partes, a criação da rede de sensores, a criação de um servidor e o desenvolvimento de uma aplicação para um "*Smartphone*".

1. Inevitavelmente, antes de iniciar o processo de desenvolvimento da arquitetura houve um processo que conduziu à seleção do material necessário. Esta seleção possibilitou ganhar consciência da dimensão do projeto e quais os pormenores a ter cuidado à aquisição de material.

2. A primeira fase da arquitetura viabilizou a construção de uma rede de sensores utilizando como recurso a tecnologia "*Mesh*"e uma outra parte da comunicação com um "*Smartphone*"utilizando a tecnologia Bluetooth.

3. A segunda fase foi a implementação de um sítio web com acesso a uma base de dados e uma base de dados. Nesta fase foram tidas em consideração questões como a segurança, o design do site, bem como a forma de apresentação dos dados sentidos pelos sensores.

4. O desenvolvimento da aplicação para "*Smartphone*"realizou a ponte entre a rede de sensores "*Mesh*"e o servidor, criando assim um meio de passagem de dados.

4.1 Arquitetura

A arquitetura desenvolvida foi implementada com a introdução de uma rede de sensores através da tecnologia "*Mesh*"e a implementação da comunicação entre um "*Hop*"e com a criação de um sistema de comunicação entre um "*Hop*"e um "*Smartphone*". Nesta fase foram desenvolvidos três "*Hops*", o "*Hop Master*", o "*Hop Slave 1*"e o "*Hop Slave 2*", onde cada um deles possui características próprias. O "*Hop Master*"tem a capacidade de gerir a rede "*Mesh*", de guardar os dados sentidos pelos sensores em um cartão SD, contabilizar o tempo e sentir a temperatura e a pressão através de um sensor. O "*Hop Slave 1*"tem a capacidade de sentir vários gases com o sensor MQ2, a humidade e a temperatura com o sensor DHT11 e a humidade do solo. O "*Hop Slave 2* tem a capacidade de sentir temperaturas através de termopares, e o gás Monóxido de Carbono através do sensor MG811. Quando ambos "*Hops*"realizam a leitura do valores sentidos por estes sensores, toda a informação é enviada para o "*Hop Master*". Durante a evolução da primeira fase da Arquitetura, foram desenvolvidos sete programas em etapas diferentes, sendo que os programas finais são "*MasterV3.ino*", "*SlaveV3.ino*"e o "*Slave2.ino*", correspondendo aos "*Hops Master*", "*Slave 1* e "*Slave 2*", respetivamente.

Adicionalmente foi criado um base de dados e um sítio permite albergar os dados sentidos pelos diversos sensores e uma interface gráfica que permite visualizar os dados através de gráficos.

Por último, foi desenvolvimento uma aplicação o sistema operativo "*Android*". Esta aplicação permite a comunicação entre o "*Hop Master*"e o "*Smartphone*"através da tecnologia Bluetooth, porém a restante comunicação ainda não está implementada mas existe o compromisso de a terminar até à defesa desta tese. A comunicação, ainda por desenvolver, consiste em interligar a aplicação à base de dados.

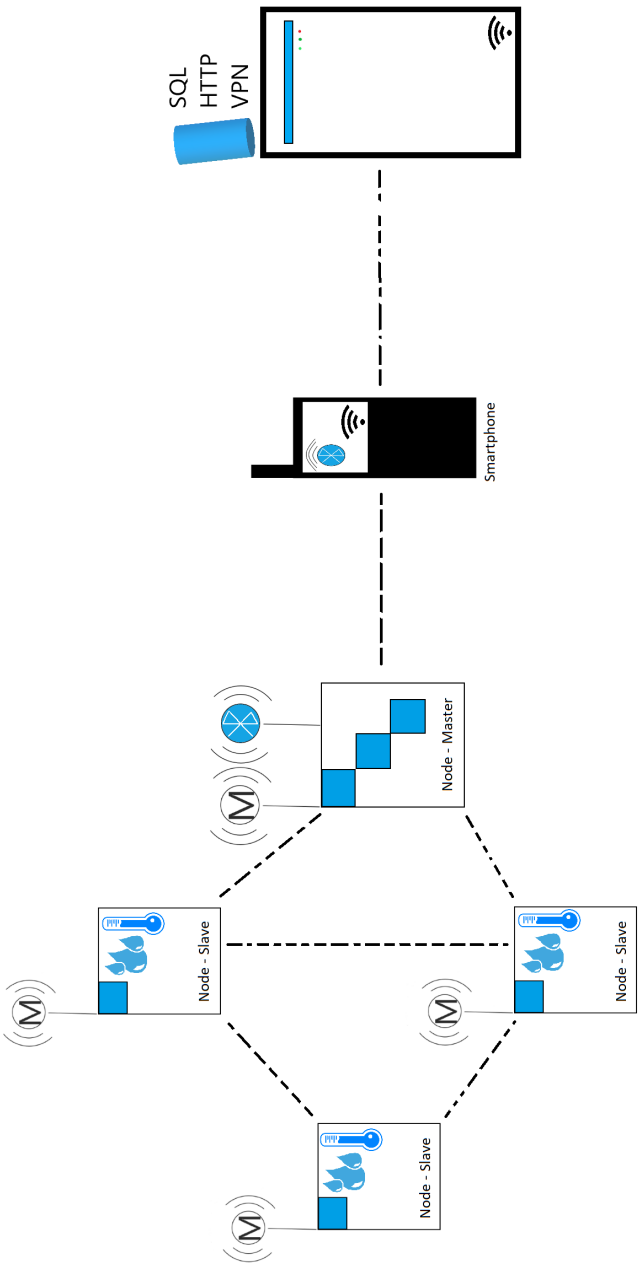

Figura 4.1: Arquitetura Implementada

4.2 Seleção do Material

A primeira fase de todas, após ter consolidado os objetivos da tese e qual a arquitetura a implementar, foi iniciar do processo de verificação do material a adquirir e a reutilizar. Esta fase passou por vários pontos de reflexão, desde quais as tecnologias de comunicação que poderiam ser usadas, os sensores que poderiam ajudar a resolver os problemas relacionados com a estufa e com a casa com a tecnologia de lama e, bem como as plataformas em que se poderá desenvolver hardware e software.

4.2.1 Micro-controlador Arduino

O primeiro aspeto foi decidir em qual plataforma se iria construir a rede de sensores. Os micro-controladores "*Arduino*"são uma boa opção pois são acessíveis, têm uma grande variedade de escolha nos modelos existentes, o desenvolvimento de um programa para um modelo é facilmente adaptado para outros modelos, a linguagem de programação usada é o C/C++ e os conteúdos são "*Open Source*". Estes últimos são desde a nível de "*Hardware*", como por exemplo o esquemático elétrico dos vários modelos do micro-controlador "Arduino", como a nível de "*software*", como por exemplo uma vasta lista de bibliotecas que permitem a interligação de vários dispositivos externos, informação exposta no capítulo "Fundamentos Tecnológicos"na secção "Micro-Controlador "*Arduino*"3.2. Uma outra razão pela qual levou a utilização da marca "*Arduino*"foi por existir um "*IDE*"que possibilitou programar e carregar os programas desenvolvidos para os micro-controladores, consulte o capítulo "Anexos"na secção "*Arduino IDE*"I.2 para mais informações.

Neste projeto de tese foram utilizados dois modelos do micro-controlador "*Arduino*", o "*Arduino Uno*"e o "*Arduino Nano*". O primeiro micro-controlador foi escolhido por já ter sido utilizado em outros projetos, as suas especificações técnicas estão no capítulo "Fundamentos Tecnológicos"na secção "Micro-controlador Arduino Uno"3.2.1. O segundo micro-controlador, o "*Arduino Nano*", serve para demonstrar que facilmente se pode adaptar o hardware e o software a um outro micro-controlador "*Arduino*". No entanto este micro-controlador é de menor dimensão do que o "*Arduino Uno*"e tem mais portas "*Input/Output*"e no capítulo "Fundamentos Tecnológicos"na secção "Micro-controlador Arduino Nano"3.2.2 encontram-se as suas especificações.

4.2.2 Comunicações

O segundo aspeto foi garantir as comunicações entre micro-controladores ou entre um micro-controlador e um "*Smartphone*". Visto que um micro-controlador contém a informação proveniente dos seus sensores, este tem de ser capaz de transmitir essa informação para outro micro-controlador, por essa razão foi decidido criar uma rede de sensores "*Mesh*".

Neste projeto de tese cada micro-controlador com vários sensores tem um módulo de comunicação "*Mesh*", que permite criar a rede de sensores e cada micro-controlador com

um rádio é considerado um "*Hop*". Porém cada micro-controlador contem um só módulo de comunicação "*Mesh*", logo a arquitetura "*Mesh*"a usar é a de primeira geração "*Radio ad hoc*", como referido no capítulo "Anexos"na secção "Mesh"I.3.1.

O módulo de comunicação usado foi o nRF24L01+. Esta placa tem baixo consumo energético, pequenas dimensões, compatibilidade com os micro-controladores da marca "Arduino"e com o mini-pc "Raspberry Pi"e a interface de comunicação é o SPI 3.2.3.1. A placa nRF24L01+ opera na banda ISM 2.4 [GHz] com uma tensão de alimentação de 3 [V], informação detalhada no capítulo "Fundamentos Tecnológicos"na secção "nRF24L01+"3.6.1. Como esta placa de comunicações opera na banda 2.4 [GHz], que é a mesma da tecnologia "*Wi-Fi*"e da tecnologia "*Bluetooth*"e por ser compatível com os micro-controladores da marca "*Arduino*"e com os mini-pc "Raspberry Pi", existem bibliotecas "*Open-Source*"que permitem tornar este módulo de comunicação compatível com as tecnologias "*Wi-Fi*"e "Bluetooth Low Energy", informação exposta no capítulo "Anexos"na secção "Comunicações: Bluetooth"I.3.2.

A comunicação realizada entre o "*Hop Master*"e o "*Smartphone*"é feita utilizando a tecnologia "*Bluetooth*"e o módulo de comunicação usado é o módulo BT HC-06, esta placa tem baixo consumo energético, é compatível e tem tamanho compacto, consulte o capítulo "Fundamentos Tecnológicos"na secção "BT HC-06"3.6.2. Esta comunicação também poderia ser feita com a tecnologia "*Bluetooth Low Energy*", através do módulo de comunicação nRF24L01+, porém nem todos os dispositivos "*Smartphones*"são compatíveis com esta tecnologia. Nesta dissertação não foi possível desenvolver esta integração por não possuir um dispositivo "*Smartphone*"compatível com esta tecnologia.

4.2.3 Sensores

A rede "*Mesh*"só se torna em uma rede de sensores quando os seus "*Hops*"contêm sensores e têm a capacidade de gerir a informação sentida. Dessa forma foram construídos dois tipos de "*Hops*", um "*Master*"e um "*Slave*". O "*Master*", para além de gerir as comunicações, também regista os dados num cartão de memória micro SD com a data e hora registada, bem como a ligação de diversos sensores. O "*Hop Slave*"apenas está ligado a vários sensores e após a captação dos dados sentidos envia essa informação para o "*Hop Master*".

As razões pelas quais levaram escolher determinado sensor dependeu das variáveis a monitorizar, do tipo de interface com o micro-controlador, preço e se existia material disponível.

Uma vez que existe o objetivo de comparar as variáveis de ambiente de uma casa com tecnologia de lama com uma casa contemporânea e monitorizar variáveis de ambiente de uma estufa, considera-se que as variáveis de interesse a monitorizar são a temperatura, a humidade, a pressão atmosférica , a qualidade do ar, os níveis de monóxido de carbono e de outros gases.

- O termopar TMP36 com o conversor A/D Max38155, consulte o capítulo "Fundamentos Tecnológicos"na secção "Max38155 com o TMP36"3.4.1. Este sensor foi escolhido por causa da sua dimensão, pois permite a inserção da ponta de prova em qualquer uma das paredes da casa com a tecnologia de lama. Com a utilização desta funcionalidade poderemos obter o valor de temperatura no interior da parede.

- O sensor de temperatura e de humidade relativa do ar DHT11, consulte no capítulo "Fundamentos Tecnológicos"na secção "DHT11"3.4.2. O sensor DHT11 foi usado para realizar a monitorização da temperatura e humidade do ar na estufa e na casa com tecnologia de lama. Visto que as estufas são longas em comprimento e normalmente têm um termómetro à porta, a temperatura sentida por esse termómetro é influenciada pela abertura da porta e pela variação da temperatura ao longo da estufa. A solução encontrada foi integrar vários "*Hops*"ao longo da estufa com o sensor DHT11 incorporado. No caso da casa com a tecnologia de lama, visto a área é cerca de 1 [m^2], não existe necessidade de utilizar mais do que um sensor DHT11 integrado em um "*Hop*"devido à sua dimensão reduzida.

- O sensor MQ2 permite avaliar a qualidade do ar e detetar e quantificar a concentração de vários gases. Os possíveis gases sujeitos a avaliação são o hidrogénio , gás de petróleo liquefeito, metano, álcool e propano, consulte o capítulo "Fundamentos Tecnológicos"na secção "MQ2"3.4.4. Este sensor de gás foi escolhido por dois motivos. A deteção dos níveis de hidrogénio, metano e propano que são gases tóxicos para o ser humano e que poderão ser formados na casa com a tecnologia de lama. A monitorização de gases que poderão afetar o crescimento das plantas presentes nas estufas.

- O sensor MG811 permite avaliar, detetar e quantificar os níveis de dióxido de carbono no ar. Este sensor também é capaz de detetar os gases etanol, monóxido de carbono e metano, não sendo capaz de os destingir entre si, consulte o capítulo "Fundamentos Tecnológicos"na secção "MG811"3.4.5. A escolha deste sensor deve-se à sua capacidade de detetar e quantificar os níveis de dióxido de carbono, que em excesso, são prejudiciais para a saúde do ser humano. Adicionalmente, este sensor foi escolhido devido à sua capacidade de utilização nas duas aplicações em estudo. No caso da casa com tecnologia de lama, é importante compreender quais os níveis de dióxido de carbono que poderão ser formados. Já no caso da estufa, é fundamental saber a concentração de dióxido de carbono no ar devido ao facto deste afetar o crescimento das plantas.O segundo motivo é para monitorizar os níveis de dióxido de carbono na estufa e mais tarde poderá ser estudado o seu efeito no crescimento das plantas.

- O sensor de Humidade do Solo deteta e quantifica os níveis de humidade do solo, todavia este sensor necessita de ser devidamente calibrado. Para tal, utiliza-se como recurso a ferramenta matemática funções de transferência previamente calibrados

em soluções aquosas conhecidas, informe-se no capítulo "Fundamentos Tecnológicos"na secção "Humidade do Solo"3.4.6. Procedeu-se à verificação no mercado dos sensores de humidade do solo disponíveis e constatou-se que o seu método de funcionamento é idêntico. Verificou-se ainda que existia a possibilidade de adquirir um sensor previamente calibrado, embora o custo associado fosse na ordem de cem vezes superior. Por este motivo, a decisão de escolha do sensor foi meramente económica visto que o adquirido oferece a solução anteriormente descrita para a sua calibração.

- O componente MPL115A2 é um sensor que permite a quantificação de valores de temperatura e de pressão, consulte o capítulo "Fundamentos Tecnológicos"na secção "MPL115A2"3.4.3. As razões que levaram à aquisição deste sensor foram o facto de comunicar via protocolo "I_2C"e permitir que o micro-controlador "Arduino *Master*"tem a capacidade de monitorização e de gestão da rede "*Mesh*". Pretende-se com este último a utilização do sensor para a validação das capacidades do "*Hop Master*". Adicionalmente este sensor foi adquirido para a monitorização da pressão atmosférica na casa com tecnologia de lama e na estufa.

4.2.4 Dispositivos Periféricos

- O módulo "*Real Time Clock*"é um componente que permite contabilizar o tempo é extremamente útil para o registo de dados, pois permite associar a marcação temporal a qualquer grandeza sentida. O "*Real Time Clock*"usado é o componente DS1307 com EEPROM, informação detalhada no capítulo "Fundamentos Tecnológicos"na secção "DS1307 com EEPROM"3.5.2.1. As razões pelas quais este componente foi o preferido em vez do DS1302 I.4.5 são o facto de incluir um conector para uma bateria, uma EEPROM para guardar dados relativos ao tempo e, além disso, o componente DS1302 necessitaria de eletrónica adicional para operar nas mesmas condições.

- Os micro-controladores "*Arduino Uno*"e "*Nano*"têm memória volátil e mantém-se a funcionar durante 36 horas, período esse que após ser ultrapassado, resulta na sua reinicialização. A reinicialização do micro-controlador leva a que todos os dados e variáveis armazenados sejam apagados, à exceção do que se encontra armazenado na memória "*flash*". A solução encontrada foi integrar um módulo leitor de cartões micro SD, pois a memória disponível para guardar dados depende da capacidade de memória do cartão micro SD, informação exposta no capítulo "Fundamentos Teórico"na secção "Módulo Leitor de Cartões micro SD"3.5.1.

4.2.5 Mini-Pc Raspberry Pi

A escolha do computador que permita correr um servidor é bastante importante. O mini-pc selecionado foi o "*Raspberry Pi B*", cujas vantagens da sua utilização passam por ser um equipamento de baixo consumo energético, com pequenas dimensões e a capacidade de

processamento disponível é a necessária para albergar um servidor. Esta informação está exposta no capítulo "Fundamentos Tecnológicos"na secção "Raspberry Pi B"3.7.1. Conteúdo a desvantagem da utilização deste equipamento é não incorporar um ecrã e um teclado, mas que foi facilmente resolvido através de uma ligação remota via protocolo "*SSH*". A informação sobre este protocolo encontra-se detalhada no capítulo "Fundamentos Tecnológicos"na secção "Protocolo *Secure Shell* (SSH)"3.7.4.

Existia ainda a disponibilidade de a utilizar um computador portátil "*Toshiba Tecra S11-11G*", no entanto não foi escolhido por possuir uma grande capacidade de processamento, por ser de grandes dimensões quanto comparado com a solução utilizada e pelo facto do consumo energético ser superior.

4.3 Primeira Fase da Arquitetura - Montagem de uma Rede de Sensores

Conforme mencionado, a arquitetura a implementar foi dividida em três partes. Numa primeira fase, interessa estabelecer a comunicação entre "*Hops*"e entre um "*Hop Master*"e um "*Smartphone*". Após as comunicações estarem estabelecidas, passou-se para o estudo e integração de vários sensores e dispositivos periféricos na arquitetura. Os objetivos para a implementação, além das questões, são tornar a arquitetura escalável, com baixo consumo energético e com capacidade de partilha de dados entre "*Hops*". Pretende ainda resolver os problemas associados à casa com a tecnologia de lama e à estufa.

As tecnologias de comunicação usadas para a implementação são "*Mesh*", "*Bluetooth*", SPI e o I_2C. A Figura4.2 é uma representação daquilo que se pretende desenvolver nesta secção.

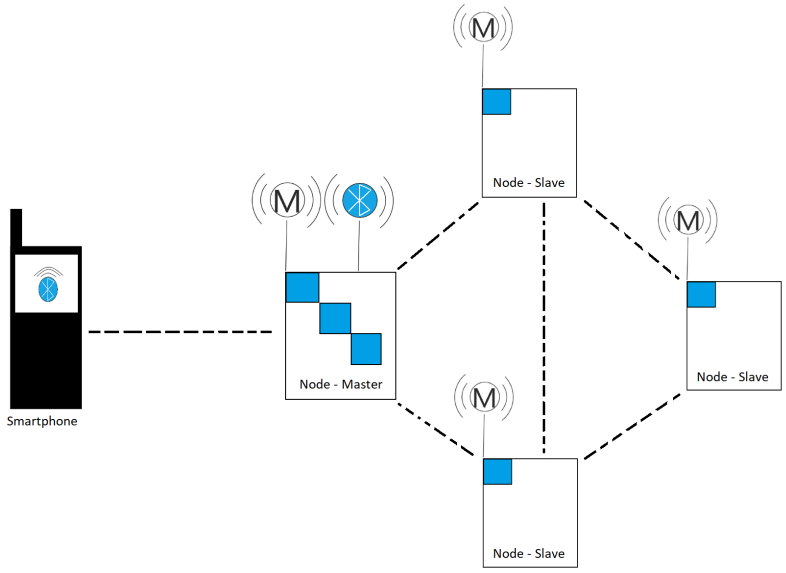

Figura 4.2: Rede Mesh implementada através da tecnologia Bluetooth de um "smartphone"

4.3.1 Módulo de comunicação nRF24L01+

A integração do módulo de comunicação nRF24L01+ 3.6.1 passou por duas fases, a primeira foi a integração do hardware em um micro-controlador "Arduino". A segunda fase foi a criação de um algoritmo que permitisse a criação de uma rede "*Mesh*"e a sua gestão. Consequentemente, criou-se um standard de comunicação entre "*Hops*"por norma a estabelecer um padrão que pudesse ser utilizado em qualquer situação.

4.3.1.1 Integração com o micro-controlador Arduino

O módulo de comunicações nRF24L01+ não vinha preparado para a integração com o micro-controlador. Esta não era expedita, pois os conectores não se encontravam soldados e nem com um cabo próprio que facilitasse a sua integração. Procedeu-se à soldadura dos conectores a dois módulos de comunicação e criaram-se dois cabos que permitissem a ligação dos terminais dos módulo aos micro-controladores, consulte a Tabela 3.28 presente no capítulo "Fundamentos Tecnológicos"na secção "nRF24l01+"no tema "Interface com o micro-controlador Arduino"3.6.1.1.

O passo seguinte foi testar na aplicação "*Arduino IDE*"se o módulo de comunicação nRF24L01+ funcionava. Isto é, através do auxílio dos programas "hellworld_rx.ino"e "helloworld_tx.ino", fornecidos pela biblioteca "*RF24Network.h*", foram carregados para os micro-controladores. O programa "hellworld_rx.ino"torna um micro-controlador num recetor "*Hop*"e está dividido em duas partes. A primeira corresponde a criação de uma estrutura de dados que permite o acesso aos valores recebidos e que seja comum ao programa "*helloworld_tx.ino*", informação presente no capítulo "Fundamentos Teóricos no tema "Interface com o Micro-controlador Arduino"na secção "nRF24L01+"no §2 3.1. A segunda parte consiste em ler os dados recebidos e escrever na estrutura de dados anterior. O acesso às mensagens recebidas é realizada através do uso de funções que permitem a impressão dos dados com a funcionalidade "*Serial Monitor*"na aplicação "Arduino IDE".

O programa "helloworld_tx.ino", que torna o micro-controlador em um transmissor, também está dividido em duas partes. Uma primeira que consiste na estrutura de dados, representada na listagem 3.1, e uma segunda para o envio de uma mensagem para o "*Hop*"recetor.

Ambos os programas poderemos observar que existem partes do código que são idênticas, como por exemplo, a estrutura de dados 3.1, a forma como o endereço de cada "*Hop*"é registado I.9 e a função que permite a sincronização da informação I.10.

Após os conhecimentos adquiridos do capítulo "Fundamentos Tecnológicos"na secção "nRF24L01+"3.6.1, mais o conhecimento ainda agora referidos, realizou-se o teste ao hardware, por forma a garantir o seu correto funcionamento. Durante o teste, reparou-se que o tempo de estabelecimento de uma comunicação era excessivo e a taxa de sucesso de envio de pacotes era baixa. Concluiu-se que o baixo rácio de transmissão de dados devia-se a uma alimentação intermitente do módulo. Para contornar este insucesso, a solução adotada foi ligar um condensador aos terminais de "GND"e "V_{CC}".

A determinação de qual o condensador ideal para estabilizar a alimentação e, consequentemente, melhorar a taxa de transmissão, levou ao ensaio de vários condensadores. Para tal foram testados condensadores de 1 [nF], 10 [nF], 10 [μF] e 470 [μF]. Inicio-se o teste com a utilização do condensador de 1 [nF] e os resultados obtidos foram que o condensador aplicado não possuía capacidade suficiente para retificar a alimentação. Após o insucesso com o primeiro condensador, procedeu-se ao teste com o condensador de 10 [nF]. Observou-se que a taxa de envio de pacotes manteve-se inalterada, não solucionado a instabilidade na alimentação do módulo. Efetuou-se uma terceira tentativa com um condensador de 10 [μF]. Este apresentava melhorias significativas em relação aos anteriores na taxa transmissão e na estabilidade de alimentação. No entanto, apesar das melhorias obtidas, foi testado o último condensador. Verificou-se que a prestação da comunicação de dados e estabilidade de alimentação manteve-se inalterada com aplicação de um condensador de 470 [μF]. Por este motivo, foram aplicados os condensadores de 10[μF] e 470 [μF] através da utilização de técnicas de soldadura nos cabos.

Após os problemas de alimentação do módulo terem sido resolvidos deu-se o início ao desenvolvimento do algoritmo que permite a comunicação entre "*Hops*"numa rede "*Mesh*". Para tal foram desenvolvidos os programas "MasterV1.ino"e o "SlaveV1.ino", onde as funções de transmissão e receção são alocada nos mesmos programas

O programa "MasterV1.ino", representado pela Figura I.14, tem o objetivo de gerir a rede "*Mesh*"enquanto que o programa "SlaveV1.ino"tem os objetivos de gerir vários sensores e enviar a informação sentida para o "*Hop Master*. Após a criação destes programas, foram criadas vários tipos de mensagens que podem ser trocados entre "*Hops*"e vários modos de funcionamento de micro-controladores.

Tabela 4.1: Tipos de mensagens entre Hops V1

Comando	Descrição	Modo	Direção
H	Comando Hello	registationMode	Slave -> Master
T	Comando Transmiting	receiverMode	Master -> Smartphone
A	Comando Accept	registrationMode	Master -> Slave
D	Comando Denial	registrationMode	Master -> Slave
R	Comando Request	receivingDataMode	Master -> Slave

Na Tabela 4.1, o comando "*Hello*"(H) permite o registo de um novo "*Hop Slave*"na rede "*Mesh*"e os novos registos são geridos pelo "*Hop Master*". Este, após a avaliação do registo, envia um comando "*Accpet*"(A) ou um comando "*Denial*"(D), consoante se o registo foi feito com sucesso ou não, respetivamente.

O comando "*Transmitting*"(T) permite enviar a informação dos sensores para o dispositivo smartphone através da tecnologia Bluetooth. Este comando só é ativo após o comando "*Request*"(R) ter sido chamado e consiste num pedido de dados pelo "*Smartphone*". O pedido de dados ocorre em dois momentos - quando o "*Smartphone*"pretende receber os dados dos sensores, processo simulado por um botão, e quando o "*Hop Master*"requisita

aos diversos "*Hop Slaves*"existentes na rede "*Mesh*".

A estrutura geral do programa "MasterV1.ino"é representado pelo o fluxograma na Figura I.14 e consiste nos processos de comunicações, receção e transmissão de mensagens.

O processo de comunicações, representado pela Figura I.15, consiste na inicialização da comunicação série, inicialização do protocolo de comunicação SPI, inicialização do módulo de comunicações nRF24L01+ e a criação da rede "*Mesh*". Em paralelo, após a ativação da comunicação série é executada a função "*SerialEvent()*"que permite avaliar as comunicações que são realizadas na comunicação série. Neste programa apenas o caracter "R"é reconhecido e é efetuado a ativação do modo de pedido de dados dos sensores.

O processo de receção de mensagens, representado pela Figura 4.3, é gerido reconhecendo os vários comandos predefinidos na Tabela 4.1. O comando H permite a ativação do modo de registo de um "*Hop*"e o comando T permite a ativação do modo de transmissão.

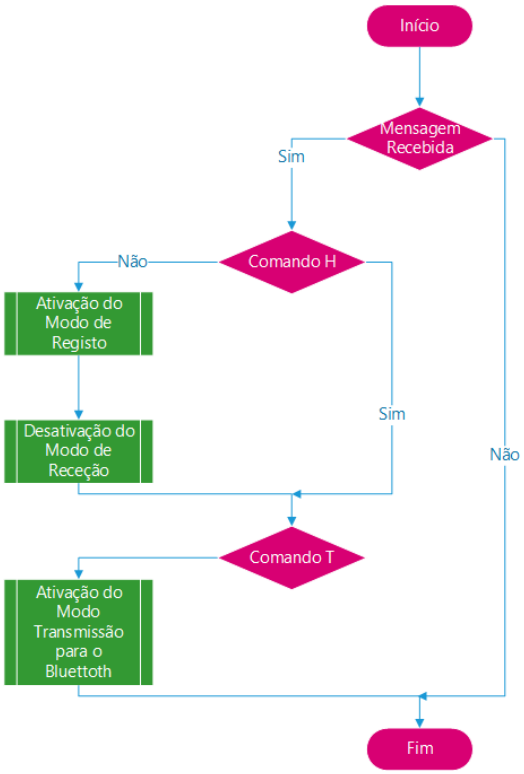

Figura 4.3: Representação do processo de receção de mensagens do programa MasterV1.ino por um fluxograma

O processo de transmissão de dados, representado pela Figura 4.4, está dividido em duas partes. A primeira parte é a execução do modo de registo, em que um "*Hop Slave* ao enviar uma mensagem com o comando H, é realizado uma verificação do registo do seu endereço. Caso já tenha sido, registado é enviado uma mensagem com o comando D, que indica a falha no registo. Caso contrário, o "*Hop Slave*"é registado e é enviada uma mensagem com o comando A, que indica o sucesso de registo do seu pedido.

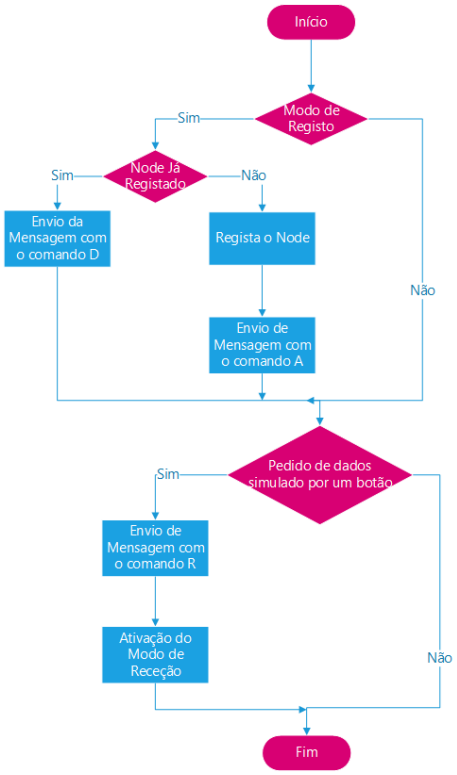

Figura 4.4: Representação do processo de transmissão de mensagens do programa MasterV1.ino por um fluxograma

Relativamente ao processo de envio de uma mensagem, este inicia-se com o envio de uma mensagem seguido da verificação da chegada da mensagem ao destino. Caso não tenha chegado ao destino, a mensagem é enviada novamente até a sua receção bem sucedida. Caso contrário, termina o processo de envio de mensagem. A Figura I.16, é uma representação do processo de envio de uma mensagem por fluxograma.

O programa que gere o "*Hop Slave*"é o "SlaveV1.ino", cujo algoritmo funciona de forma semelhante ao do programa "MasterV1.ino". No entanto, apenas apresenta ligeiras modificações nos modos de receção, transmissão e foi adicionado um processo Timer. A representação geral do programa "SlaveV1.ino"é apresentado pelo fluxograma na Figura I.17.

O processo de comunicações, representado pela Figura 4.5, apresenta duas funcionalidades que são executadas em paralelo a partir da ativação da comunicação série. A primeira funcionalidade contém a inicialização da comunicação série, da comunicação SPI, módulo nRF24L01+ e da criação da rede "*Mesh*". A segunda funcionalidade apenas gere as comunicações série quando são capturados os caracteres "s"ou "S", "b"ou "B"e "r"ou "R". Após a captura de algum dos caracteres mencionados, são executados outros processos. Adicionalmente, sempre que são capturados os caracteres "b"ou "B"é ativada a funcionalidade "*debug*"do programa. Esta funcionalidade permite a impressão de vários valores de interesse para melhor compreender as várias fases do programa e aonde se encontra as possíveis falhas.

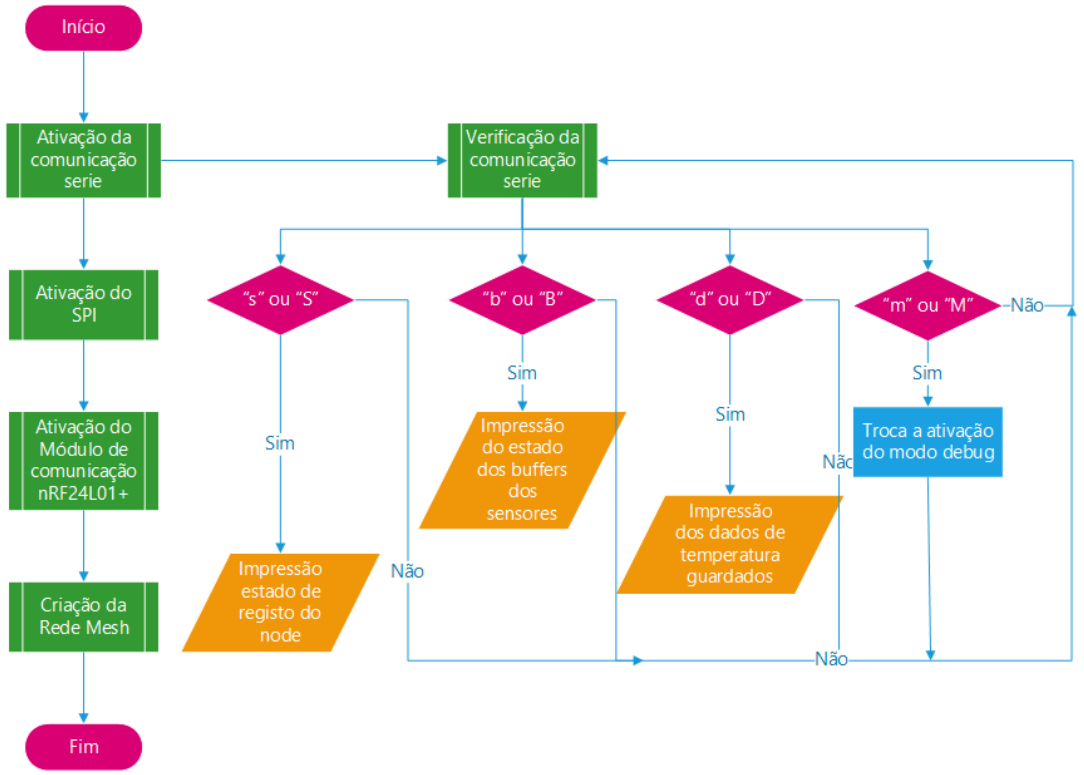

Figura 4.5: Representação do processo de comunicações do programa SlaveV1.ino por um fluxograma

O processo "*Timer*"I.18, por defeito, apenas é executado com o intervalo de 30 minutos. Quando este é executado, é realizada a recolha de informação sentida pelos sensores. Tomou-se em consideração, neste caso, que nenhum sensor estava conectado ao micro-controlador e, por essa razão, um sensor de temperatura foi simulado especificando um valor de temperatura de 22 [°C].

O processo de receção de mensagens é representado pelo fluxograma ,representado pela Figura 4.6. Neste processo são executadas três funcionalidades. São elas, o comando A, quando é detetada uma mensagem que permite indicar ao "*Hop Slave*"o sucesso do seu pedido de registo, o comando D, que apenas imprime no "Serial Monitor"o caracter "D"e o comando R, que permite a ativação do modo de envio de dados dos sensores.

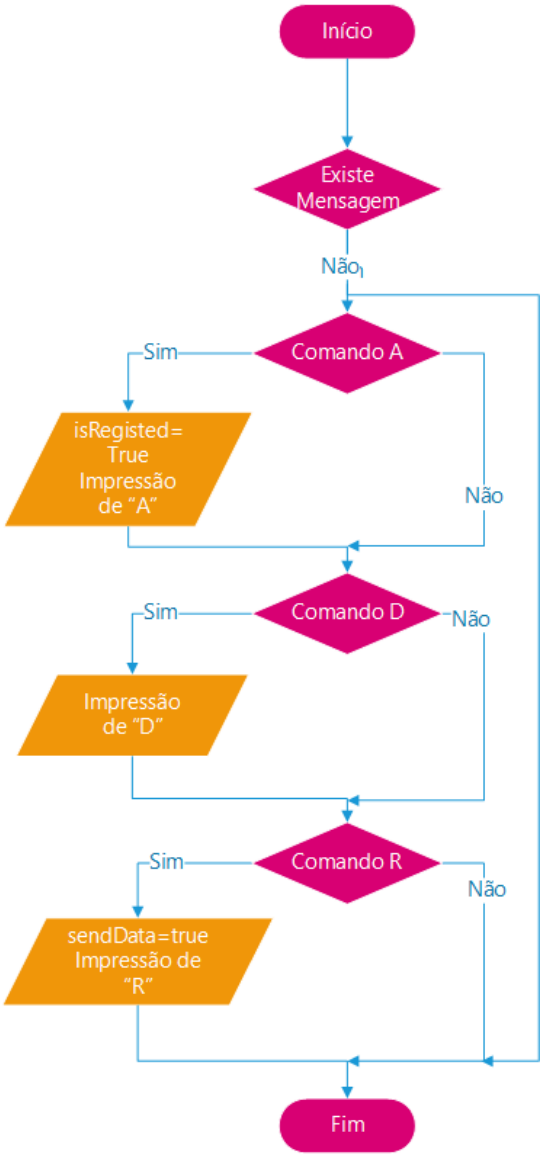

Figura 4.6: Representação do processo de receção de mensagens do programa SlaveV1.ino por um fluxograma

O processo de transmissão de dados está representado pelo fluxograma na Figura 4.7. Este processo contém duas funcionalidades; a primeira permite o pedido de registo do Hop Slave no Hop Master e a segundo é permite o envio dos "*buffers*"com os dados dos sensores. Neste caso apenas são enviados os valores simulados de um sensor de temperatura.

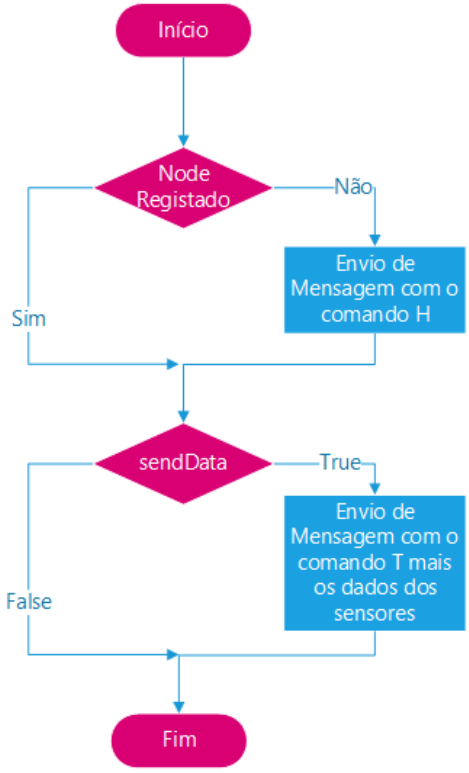

Figura 4.7: Representação do processo de transmissão de mensagens do programa SlaveV1.ino por um fluxograma

As comunicações efetuadas entre os "*Hops*"e entre um "*Smartphone*"é representado pelos diagramas de sequência UML. Aonde o primeiro 4.8 consiste na representação das comunicações quando o "*Hop Slave*"é registado com sucesso e o segundo é a representação das comunicações quando o Hop Slave não é registado com sucesso. 4.9.

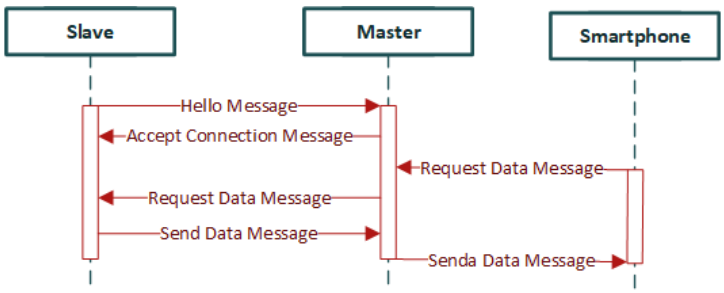

Figura 4.8: Representação das comunicações entre Hops e smartphone por um diagrama de sequência quando é aceite o registo do Node Slave

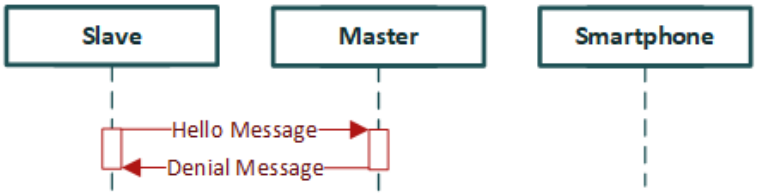

Figura 4.9: Representação das comunicações entre Hops e smartphone por um diagrama de sequência quando não é aceite o registo do Hop Slave

O passo seguinte, que se encontra nas secções 4.3.2 4.3.3 4.3.4 deste capítulo , foi a adição de sensores à arquitetura e foram criados os programas "*MasterV3.ino*"e o "*SlaveV3.ino*".

Ao ser adicionado o módulo de leitor de cartões, o paradigma de comunicações mudou pois o "*Hop Master*"possuía mais capacidade para guardar informação sentida pelos sensores. Assumiu-se inicialmente, que a informação sentida por cada "*Hop*"era armazenada na sua memória volátil e tal foi alterado porque o "*Hop Master*"possui uma maior capacidade de armazenamento de dados em memória não volátil. Como consequência foram desenvolvido os programas "*MasterV3.ino*"e "*SlaveV3.ino*". O paradigma ocorre quando um "*Hop Slave*"tem uma nova informação sentida pelo sensor e este "*Hop*"envia essa informação de imediato para o "*Hop Master*. Como este "*Hop*"contém também um módulo de "*Real Time Clock*", um barómetro e um termómetro, a nova informação é guardada com data, hora, temperatura e pressão atmosférica no cartão "*microSD*". Para além dos dados dos sensores, também é guardado o endereço de origem da mensagem.

Para corresponder a nova estrutura de comunicações, foi adicionado mais um comando para as mensagens trocadas na rede "*Mesh*", em que a Tabela 4.2 demonstra as diferenças dos comandos das mensagens.

Tabela 4.2: Tipos de mensagens entre Hops V2

Comando	Descrição	Modo	Direção
H	Comando Hello	registationMode	Slave -> Master
T	Comando Transmitting	receiverMode	Master -> Smartphone
A	Comando Accept	registrationMode	Master -> Slave
D	Comando Denial	registrationMode	Master -> Slave
R	Comando Request	receivingDataMode	Master -> Slave
S	Comandos Send	transmittingMode	Master-> Smartphone e Slave -> Master

Comparando a Tabela: 4.2 com a Tabela; 4.1, é possível constatar que foi introduzido um novo comando. O comando S permite que seja enviada uma mensagem com os dados dos sensores. A comunicação pode ser realizada entre um "*Hop Master*"e um "*Smartphone*"ou entre um "*Hop Slave*"e o "*Hop Master*". O comando T anteriormente tinha a mesma função que o comando S, porém agora o comando T apenas indica quantas mensagens irão ser recebidas com o comando S.

A representação geral do programa "*MasterV3.ino*"é apresentado pelo fluxograma na Figura I.14. No entanto existem alterações no processo de comunicações e receção e transmissão de mensagens. O processo de comunicações é representado pelo fluxograma na Figura 4.10. A única alteração neste processo foi a adição de um ciclo que não permite a inicialização de nenhum outro sub-processo enquanto a comunicação série não é estabelecida.

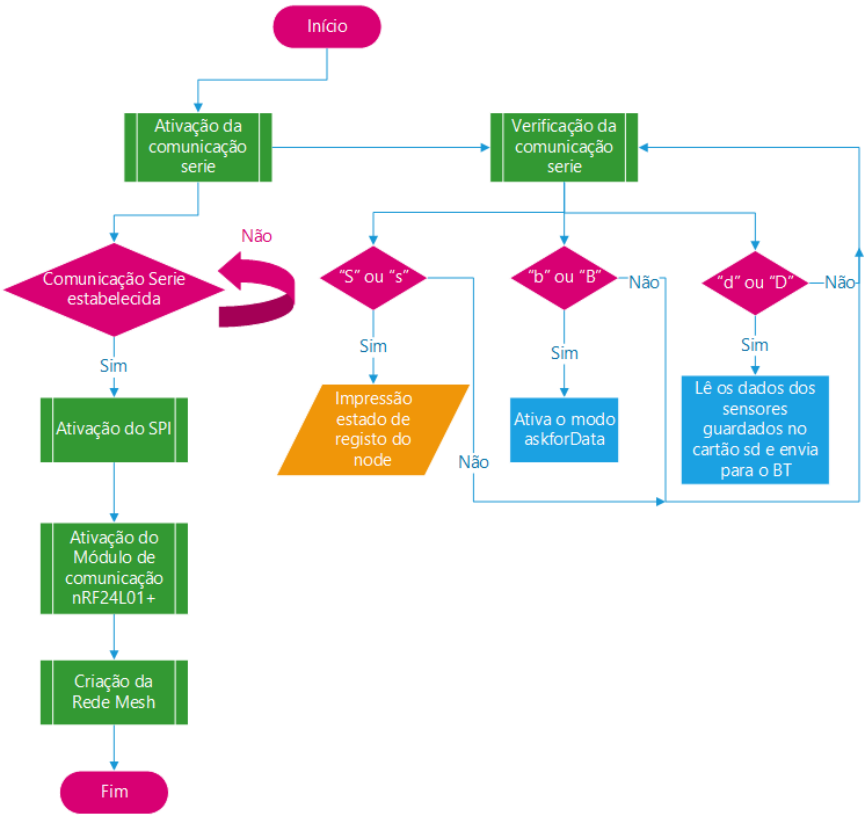

Figura 4.10: Representação do processo das comunicações do programa "MasterV3.ino"

O processo de receção é representado pelo fluxograma na Figura 4.11. As alterações em relação ao programa "*MasterV1.ino*"são a mensagem com o comando T que apenas contém a informação de quantas mensagens irão ser transmitidas e uma nova mensagem com o comando S, que transmite a informação sentida pelos sensores. Quando esta mensagem é recebida a informação é guardada em um ficheiro. Para além destes comandos foi adicionado um modo, que permite quando um "*Smartphone*"faz um pedido de dados, este recebe os dados dos sensores de outros "*Hops*"e envia para o "*Smartphone*"via Bluetooth.

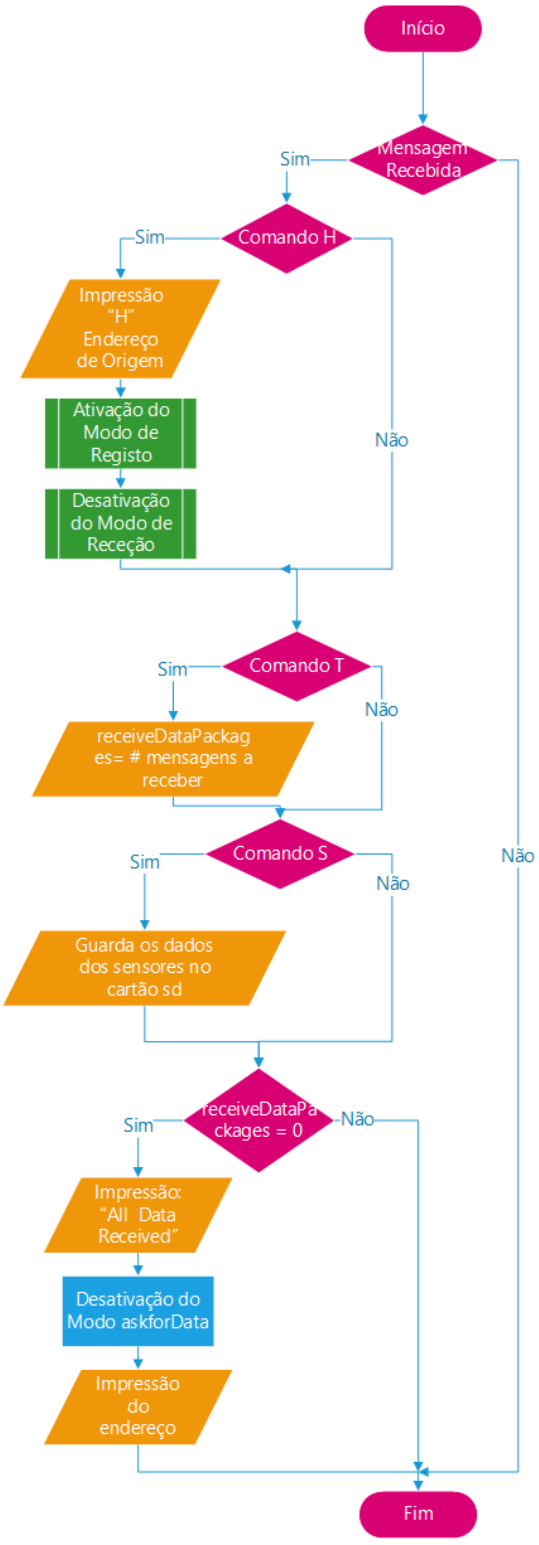

Figura 4.11: Representação do processo de receção de mensagens do programa "MasterV3.ino"

O processo de transmissão é representado pelo fluxograma na Figura 4.12. Neste processo foi adicionado a funcionalidade de enviar o registo de hora e data para os "*Hops Slave*".

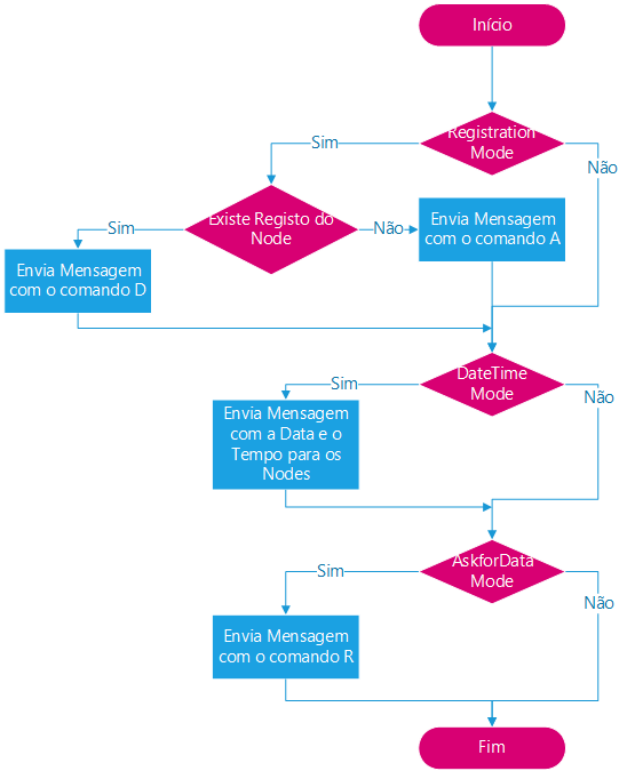

Figura 4.12: Representação do processo de transmissão de mensagens do programa "MasterV3.ino"

O novo paradigma de comunicações é representado pelo diagrama de sequência UML 4.13, no entanto apenas se verifica quando o registo "*Hop Slave*"é efetuado com sucesso, caso contrário a sua representação está apresentada no diagrama 4.9

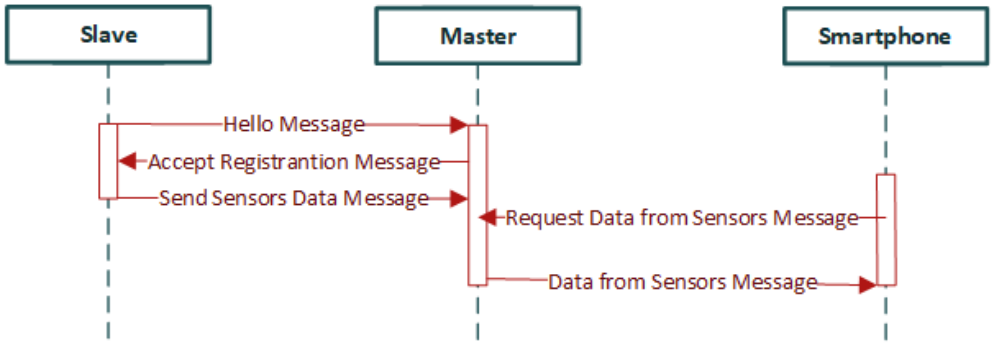

Figura 4.13: Representação do novo paradigma de comunicações por um diagrama de sequência UML

4.3.2 DHT11

O sensor DHT11 3.4.2, antes de ser integrado na arquitetura, foi integrado num outro micro-controlador para que fosse possível estudar o seu funcionamento e verificar o seu correto funcionamento.

4.3.2.1 Integração no micro-controlador "Arduino"

A partir dos conhecimentos adquiridos no capítulo "Fundamentos Tecnológicos"na secção "DHT11"3.4.2, a integração do hardware foi bastante simples. Com a informação disponibilizada na Tabela: 3.12 presente no capítulo "Fundamentos Tecnológicos"na secção "DHT11"no tema "Especificações Técnicas", a integração deste sensor no micro-controlador "Arduino"foi realizada através da ligação de terminais de acordo com a Tabela: 4.3.

Tabela 4.3: Ligação dos terminais dos sensores aos terminais Arduino

Terminal do Sensor	**Terminal Arduino**
V_{CC}	5V
S	8
GND	GND

O software disponibilizado neste sensor para a integração no micro-controlador implicava a adição da biblioteca "DHT.h"ao projeto. No entanto, para melhor compreender a forma como o sensor DHT11 operava, foi usado o programa "*DHTTester.ino*", disponibilizado com a biblioteca. Com este programa, foi entendido que é necessário escolher qual o tipo do sensor DHT que se está a utilizar e quais as funções que permitem ler os dados sentidos pelo sensor.

Neste caso o tipo do sensor escolhido foi o DHT11, embora para escolher um dos outros tipos, basta utilizar a linha de código associada. As funções que permitem o acesso aos valores de temperatura e de humidade sentidos pelo sensor são as descritas na listagem I.12.

No algoritmo representado pela listagem I.12, podemos identificar que as funções de interesse tais como "*dht.readHumidity()*", devolve o valor da humidade relativa sentido pelo sensor e a "*dht.readTemperature()*"devolve o valor da temperatura sentida pelo sensor em [°C]. Caso a função "*dht.readTemperature(true)*"adote o parâmetro de entrada "True"devolve o valor de temperatura expressa em [°F]. Existe ainda a possibilidade de converter valores de temperatura para outro tipo de unidades, com recurso à função "*dht.computeHeatIndex()*"caso o parâmetro de entrada seja (f,h) a temperatura é convertida e expressa em [°F] e caso o parâmetro de entrada seja (t,h,false) o valor de temperatura é convertido e expresso em [°C].

4.3.2.2 Integração na Arquitetura

O estudo efetuado no capítulo "Fundamentos Tecnológicos"na secção "DHT11", juntamente com os conhecimentos adquiridos na fase de integração em um micro-controlador à parte da arquitetura 4.3.2.1, é possível integrar o sensor DHT11 na Arquitetura. Para tal, o programa desenvolvido foi o "*SlaveV2.ino*"e foi adicionada a biblioteca "DHT.h"ao projeto, bem como o processo "*Timer*"para as funções necessárias à recolha de informação em relativamente à temperatura e humidade. A representação do algoritmo que permite a recolha da informação sentida pelo sensor está exposto no seguinte fluxograma na Figura I.19

4.3.3 MQ2

Antes de ser integrado na Arquitetura, o sensor MQ2, foi integrado no micro-controlador "*Arduino*"para melhor compreender o seu funcionamento antecipadamente. Por forma a aferir o seu correto funcionamento deste sensor foi desenvolvido na aplicação "*Arduino IDE*"um programa que permite a leitura dos valores sentidos pelo sensor. Uma primeira aproximação revelou que a medida retornada pelo sensor não era expressa em tensão [V] pelo que foi necessário adequar o mesmo. Para tal, converteu-se a medida para um valor de tensão [V] 4.2. Visto pretender-se obter a concentração dos diversos gases que este sensor é capaz de detetar, foi iniciado o processo de estudo de calibração para o mesmo. Após o término do estudo foi desenvolvido um algoritmo que permite a realização da calibração para o sensor. Finalmente este sensor foi adicionado à Arquitetura.

4.3.3.1 Calibração e Integração com o micro-controlador "Arduino"

O estudo que se segue tem por base os conteúdos apresentados no capítulo "Fundamentos Tecnológicos"na secção "MQ2"3.4.4.

O sensor MQ2 permite quantificar a concentração e detetar sete gases. Para o efeito foi necessário calcular sete funções de transferência. Foi gerado um gráfico por cada gás através da recolha de oito pontos de cada gás da Figura 3.4, recorrendo à aplicação "*Excel*". Através da ferramenta de criação de linhas de tendência, foram calculadas todas as funções de transferência tendo em consideração que o erro quadrático de cada uma das funções deve ser o mais aproximadamente da unidade, diminuindo o erro a função transferência gerada e a sua linearização.

A Tabela 4.11 apresenta as funções de transferência calculadas com a ferramenta Excel.

Para testar as novas funções de transferência foi necessário realizar a integração do sensor MQ2 com o micro controlador "*Arduino Uno Rev 3*". A integração por hardware passa por conectar os terminais do sensor MQ2 aos terminais do micro-controlador.

Após a integração do sensor com o micro-controlador foi calculada a resistência do sensor (R_{S_AR}), sabendo que:

$$\frac{R_{S\_AR}}{R_O} = 10 \tag{4.1}$$

Tabela 4.4: Funções de Transferências dos gases do Sensor MQ2

Gás	Função de Transferência [ppm]	Erro Quadrático R^2	Linha de Tendência
Ar	-	-	-
C_3H_8	$654.290\frac{R_S}{R_O}^{-2.072}$	0.9975	Potência
Alcool C_2H_6O	$3678\frac{R_S}{R_O}^{-2.67}$	0.9934	Potência
CO	$37143\frac{R_S}{R_O}^{-3.178}$	0.9948	Potência
CH_4	$4327.7\frac{R_S}{R_O}^{-2.672}$	0.9859	Potência
GPL	$12613e^{-2.882\frac{R_S}{R_O}}$	0.9481	Exponencial
H_2	$1016.8\frac{R_S}{R_O}^{-2.091}$	0.9975	Potência

Tabela 4.5: Ligação dos terminais do sensor MQ2 ao micro-controlador "*Arduino*"

Terminal do Sensor "MQ2"	Terminal micro-controlador "Arduino"
5V	V_{CC} [V]
GND	GND
NC	-
Sinal	A0

Converteu-se o valor de saída do sensor para uma valor de tensão.

$$V_{RL} = \frac{analogRead(A0)}{Preciso} * V_{CC} \tag{4.2}$$

Em que "*analogRead(A0)*"é o comando que permite fazer a leitura do valor do sensor, a variável Precisão consiste na precisão da porta analógica, que é igual a 1024, e a variável V_{CC} corresponde a tensão de operação, que é igual a 5.0 [V].

Foi desenvolvida a equação 3.3, com o propósito de calcular a resistência do gás Ar R_S.

$$R_S = R_L[\frac{V_{cc} - V_{RL}}{V_{RL}}] \tag{4.3}$$

A Equação 4.4 é o desenvolvimento da Equação 4.3, omitindo a resistência de carga pois este parâmetro é de uma unidade expressa em [Ω].

$$R_S = \frac{V_{cc} - V_{RL}}{V_{RL}} \tag{4.4}$$

A resistência de carga V_{RL} é ajustável, pelo que neste caso está ajustada para unidade. Especificando a equação 4.4 para o gás ar, fica:

$$R_{S\_AR} = \frac{V_{cc} - V_{RL}}{V_{RL}} \tag{4.5}$$

O último passo é calcular a resistência do sensor MQ2, sabendo que 4.1 então:

$$R_O = \frac{R_{S\_AR}}{10} \tag{4.6}$$

$$R_O = \frac{V_{cc} - V_{RL}}{10 * V_{RL}} \tag{4.7}$$

Após o estudo sobre como calibrar e quanto quantificar a concentração dos diversos gases que o sensor MQ2 é capaz de detetar, foram criados dois programas na aplicação "*Arduino IDE*". Um primeiro programa, "MQ2_calibration.ino", que permite calcular a resistência do sensor R_O para que o mesmo possa ser devidamente calibrado , e um segundo programa, "*MQ2_gas_detector.ino*", que traduz a implementação da função de transferência do monóxido de carbono.

O algoritmo do programa "MQ2_calibration.ino"é representado pelo o fluxograma na Figura I.20. Iniciou-se o teste do sensor procedendo-se ao aquecimento prévio durante 2 horas. Após alcançar a referida condição o valor de resistência do sensor obtido foi de 0.49 [Ω]. No diagrama fluxograma, representado pela Figura I.20, podemos observar que é efetuada uma média dos valores retirados do sensor, precedido da conversão para valores de tensão [V], finalizando com o cálculo da resistência do ar utilizando a Equação 4.5. Por último, recorrendo a Equação 4.7, foi calculada a resistência do sensor.

Após o cálculo do valor da resistência do ar ter sido concluído, foi desenvolvido o programa "MQ2_gas_detection.ino"que permite quantificar os níveis de concentração do gás monóxido de carbono. O fluxograma, representado pela Figura I.21, é uma representação do algoritmo. As diferenças entre este programa e o anterior são: o cálculo da média de 100 valores sentidos por o sensor foi substituído apenas por uma leitura e foram retiradas as Equação 4.5 4.7 sendo substituídas pela Equação I.13.

Após o programa estar concluído, iniciou-se a fase de testes. O teste consiste em ligar o micro-controlado à corrente e deixar que o sensor MQ2 aqueça durante duas horas. Ao terminar o período de aquecimento foram calculados os valores de resistência, o rácio $\frac{R_S}{R_O}$ e os níveis de concentração de gás monóxido de carbono na atmosfera.

Tabela 4.6: Teste do programa "*MQ2_gas_detector.ino*"

# Leitura	V_{RL} **[V]**	$\frac{R_S}{R_O}$	**Concentração [ppm]**
1	1.17	6.60	92.54
2	1.17	6.60	92.54
3	1.17	6.60	92.54
4	1.17	6.60	92.54
5	1.17	6.60	92.54

A Tabela 4.6 apresenta 5 amostras retiradas do sensor MQ2 durante 5 horas, ou seja, cada amostra foi recolhida de hora em hora. Esta Tabela apresenta que o sensor MQ2 é um sensor estável, com baixa variação ao longo do tempo, para a quantidade de gás ensaiada.

4.3.3.2 Integração na Arquitetura

Após os conhecimentos adquiridos da secção anterior 4.3.3.1 estarem consolidados, o sensor MQ2 foi integrado na arquitetura. A Tabela 4.16 mostra a forma.

Tabela 4.7: Ligação dos termais do sensor ao micro-controlador "Arduino"

Terminal do sensor de MQ2	**Terminal do micro-controlador "Arduino"**
V_{CC}	3.3V
GND	GND
S	A1

O desenvolvimento no software passou por integrar no processo do "*Timer*"do programa "*SlaveV2.ino*"uma função que permite calcular os níveis de monóxido de carbono na atmosfera. A representação dessa integração encontra-se nas Figuras I.22 e I.23, onde ocorre adição de uma função no processo de "*Timer*"e a representação da função "$MQ2_RS_gas()$", respetivamente.

4.3.4 Humidade do Solo

O sensor de humidade do solo 3.4.6, antes de ter sido integrado num micro-controlador "*Arduino*", foi submetido a um estudo referente à sua calibração para que o micro-controlador calculasse uma estimativa da quantidade de massa de água por massa de terra.

Após os obstáculos terem sido ultrapassados e os conhecimentos consolidados, deu-se a integração do sensor de humidade do solo e do sensor de gás MQ2 na Arquitetura.

4.3.4.1 Calibração e integração num micro-controlador "Arduino"

O processo de calibração de um sensor de humidade é complexo e varia sempre que o tipo de solo seja diferente. Para tal é necessário recolher uma amostra do solo e colocá-la no forno para retirar toda a humidade presente na amostra. Após a humidade removida, foi verificada uma tensão gerada pelo sensor, informe-se nos sítios [20] [24] De seguida, calculasse a densidade do solo através da verificação da massa e volume da amostra.

$$\rho = \frac{Massa}{Volume}[Kg/m^3] \tag{4.8}$$

Adiciona-se uma porção de água conhecida e medir a massa [Kg] e retira-se o valor de tensão gerado pelo sensor. Repetir este passo as vezes que forem necessárias para obter um espectro alargado de tensões geradas pelo sensor.

A densidade da água é:

$$\rho_{água} = 0.98[Kg/m^3] \tag{4.9}$$

O θ_g é a massa de água presente numa certa massa de solo.

$$\theta_g = \frac{M_{soloseco} - M_{\#mistura}}{M_{soloseco}} \tag{4.10}$$

A partir do θ_g é possível calcular o θ_v, que representa o volume de água presente num certo volume de solo.

$$\theta_v = \theta_g * \frac{\rho_{terra}}{\rho_{água}} \tag{4.11}$$

Como o $\rho_{água} = 0.98 \approx 1$ a expressão θ_v fica:

$$\theta_v = \theta_g * \rho_{terra} \tag{4.12}$$

Após o estudo sobre a calibração deste sensor, foi realizando a integração do sensor de humidade do solo num micro-controlador "*Arduino*"à parte da Arquitetura. Sabendo que a tensão de alimentação é de 3.3 [V] 3.21 e a comunicação do sensor é feita através de uma porta analógica. A Tabela: 4.16 expõe como foi realizado a ligação dos terminais do sento ao micro-controlador.

Tabela 4.8: Ligação dos termais do sensor ao micro-controlador "Arduino"

Terminal do sensor de humidade do solo	**Terminal do micro-controlador "Arduino"**
V_{CC}	3.3V
GND	GND
S	A0

De seguida, foram desenvolvidos dois programa na aplicação "*Arduino IDE*".

O programa "Soil_Moisture_GetDataValues.ino"tem como objetivo recolher 100 valores do sensor, realizar a média e converter para valores de tensão. A Figura I.25 é a representação do primeiro programa por um fluxograma.

Após o programa estar desenvolvido, foi então realizada a experiência acima descrita e os resultados obtidos são os expressos na Tabela: 4.9.

Tabela 4.9: Dados obtidos na experiência do sensor de humidade do solo

# Mistura	**Massa [Kg]**	**Analog Read**
Terra Seca	0.190	1024.000
1	0.288	600.00
2	0.378	361.29
3	0.476	287.25
4	0.574	188.40
5	0.666	172.38

De seguida foram calculados os parâmetros θ_g e θ_v de acordo com as Equação 4.10 4.12, respetivamente. Os resultados obtidos foram estão apresentados na Tabela: 4.10.

A partir da Tabela: 4.10 foi criado um gráfico de estimativa do θ_g pela tensão do sensor recorrendo da aplicação "*Excel*".

Tabela 4.10: Cálculo da tensão do sensor de humidade do solo, θ_g e θ_v

# Mistura	Tensão do Sensor	θ_g	θ_v
Terra Seca	5	0	-
1	2.90351	0.51578	16.33333
2	1.74835	0.98947	31.33333
3	1.39005	1.50526	47.66667
4	0.91170	2.02105	64.00000
5	0.83418	2.50526	79.33333

Figura 4.14: Gráfico Tensão do sensor por Massa de água numa amostra de solo

Através da ferramenta de criação de linhas de tendência foi calculada a função de transferência com um erro quadrático mais aproximado da unidade, apresentado na Tabela 4.11.

Tabela 4.11: Funções de Transferências dos gases do Sensor MQ2

Função de Transferência θ_g	**Erro Quadrático** R^2	
$0.1993 * V^2 - 1.6713 * V + 3.5082$	$1016.8 \frac{R_S}{R_O}^{-2.091}$	0.9614

O segundo programa foi implementado a função de transferência, dada pela equação Função de Transferência apresentada na Tabela 4.11 , a Figura I.25 é um fluxograma que é uma representação do programa "Soil_Moisture.ino".

4.3.4.2 Implementação na Arquitetura

A integração do sensor de humidade do solo na Arquitetura fez-se em conjunto com o sensor MQ2. No entanto, nesta secção só irá ser apresentada a integração do sensor de humidade do solo. Aplicando os conhecimentos adquiridos no capítulo "Fundamentos Tecnológicos"na secção "Sensor de Humidade do Solo"3.4.6 e os conhecimentos da secção anterior 4.3.4.1 foi feita a ligação dos terminais do sensor de humidade do solo aos terminais do micro-controlador "*Arduino Uno Rev 3*"

Tabela 4.12: Ligação dos termais do sensor ao micro-controlador "*Arduino*"

Terminal do sensor de humidade do solo	**Terminal do micro-controlador "*Arduino*"**
V_{CC}	3.3V
GND	GND
SA	A2
SD	7

Após a ligação do sensor de humidade e do sensor de gás MQ2 ter sido realizada, foi então desenvolvido o programa "SlaveV2.ino". As alterações efetuaram-se no processo de "*Timer*",isto é, foi adicionada a função "*soilMoisture()*"que permite realizar a conversão dos valores sentidos pelo sensor para uma determinada massa de água presente numa amostra de solo. O processo de "*Timer*"é representado pela Figura 4.15 e a função pelo Figura 4.16.

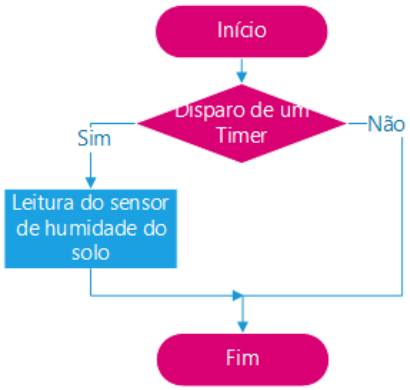

Figura 4.15: Representação do processo "*Timer*"do programa "SlaveV2.ino"por um fluxograma

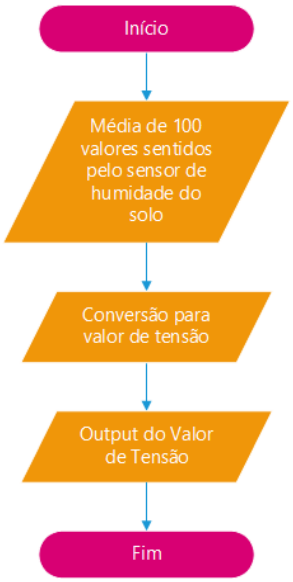

Figura 4.16: Representação da função "funcaosoilMoisture()"por um fluxograma

4.3.5 Módulo de comunicação BT HC-06

Após a integração dos sensores no "*Hop Slave*", foi integrado no "*Hop Master*"um módulo de comunicação Bluetooth BT HC-06. Neste módulo a integração foi feita na Arquitetura sem antes passar por um micro-controlador isolado da Arquitetura.

A integração deste módulo com o micro-controlador é feita realizando a ligação dos terminais corretos, reveja a Tabela 3.32 presente no capítulo "Fundamentos Tecnológicos"na secção "BT HC-06".

É importante referir que este módulo utiliza dois protocolos de comunicação. A tecnologia "Bluetooth", para fazer a ponte entre o módulo BT HC-06 com o "*Smartphone*". e a comunicação série, que permite fazer a ponte entre módulo e o micro-controlador "Arduino".

A integração deste módulo, por software, foi feita no programa "*MasterV2.ino*", utilizando a biblioteca "*SoftwareSerial.h*". Esta permite a conversão de terminais RX e TX num circuito "*UART*"parcial, realizando assim a integração do módulo Bluetooth no micro-controlador.

A representação geral do programa "*MasterV2.ino*"é apresentado pela Figura 4.17. No entanto não houve diferenças significativas neste programa em relação ao "*MasterV1.ino*", pois este já estava preparado para adicionar o módulo Bluetooth. As funções de escrita e de leitura da funcionalidade "Serial Monitor"permitem realizar a transmissão e a receção respetivamente para este módulo.

No processo de comunicações é executado uma função em paralelo, após a inicialização da comunicação série, faz a gestão da comunicação série. Visto que o módulo Bluetooth usa das funções do "Serial Monitor", esta função, com a execução em paralelo, permite capturar o comando "R"proveniente do módulo Bluetooth. A representação do

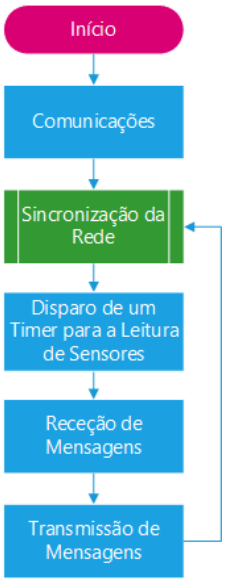

Figura 4.17: Representação geral do programa "MasterV2.ino"por um fluxograma

processo de inicialização de comunicações está apresentado na Figura 4.18.

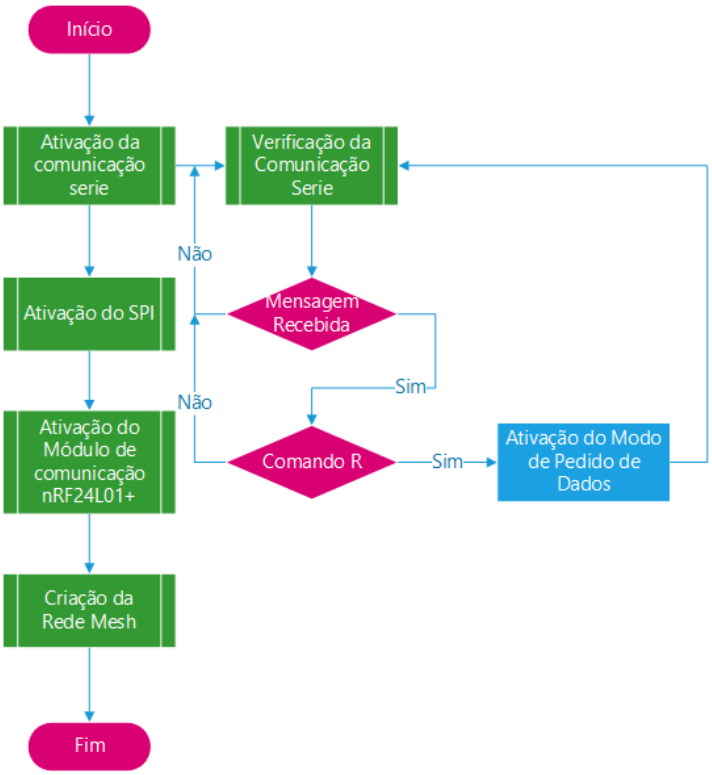

Figura 4.18: Representação do processo inicialização de comunicações por um fluxograma

A implementação por software deste módulo permitiu alterar o processo de receção de mensagens. Esta integração levou substituir a simulação de uma mensagem via Bluetooth através de um botão pela adição do módulo à Arquitetura. Durante a utilização deste programa é enviada uma mensagem pelo módulo para verificação do seu correto funcionamento. O processo de receção de mensagens está representado pela Figura 4.19.

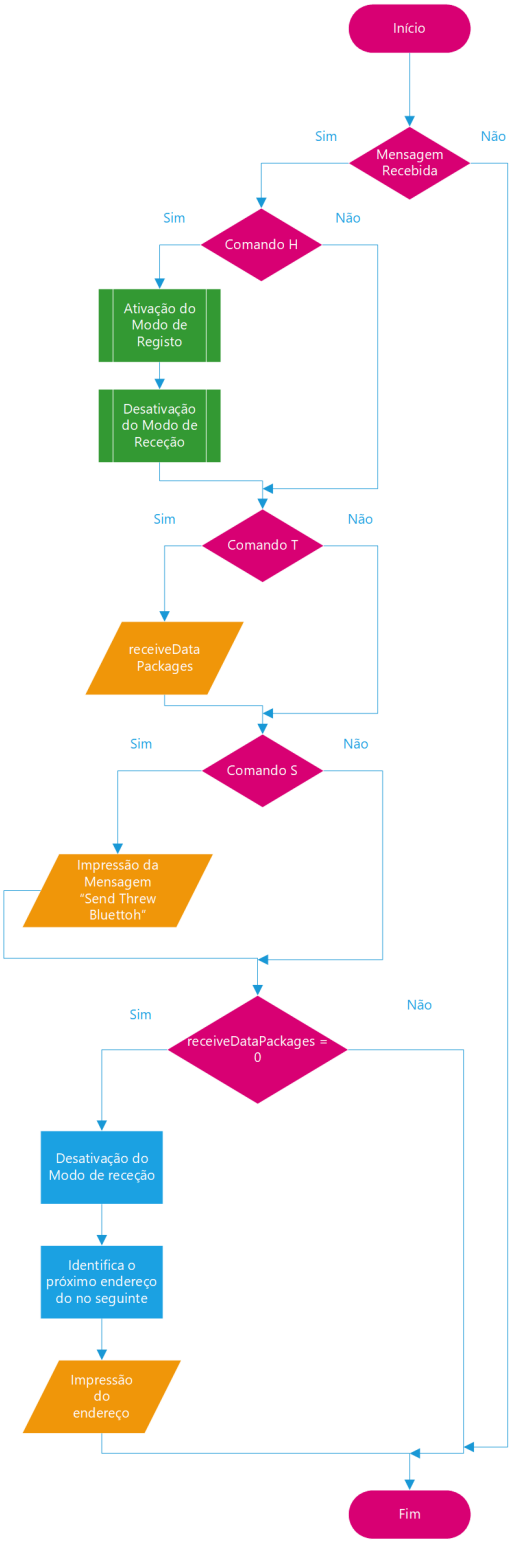

Figura 4.19: Representação do processo de receção de mensagens por um fluxograma

O processo de transmissão está representado pela Figura 4.20.

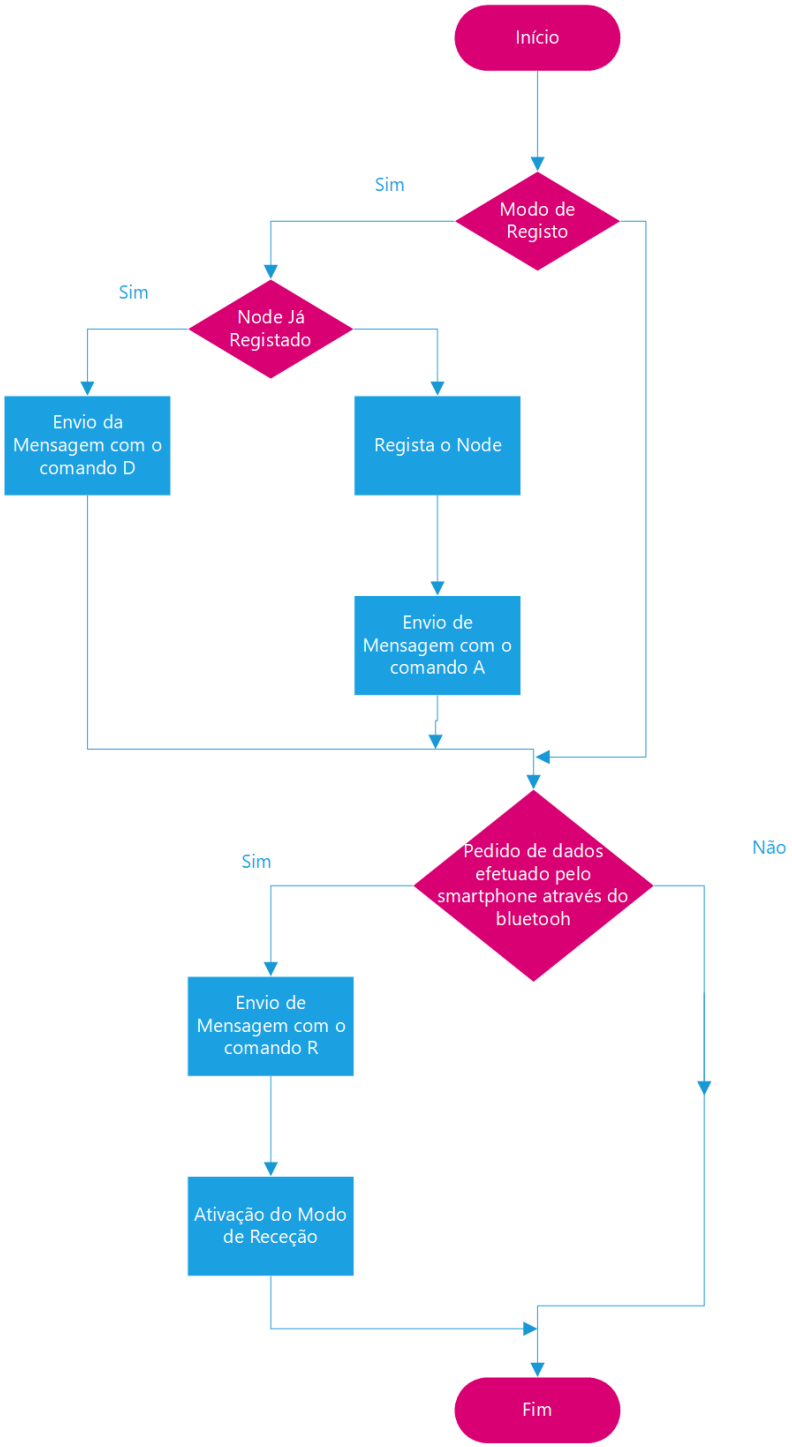

Figura 4.20: Representação do processo de transmissão de mensagens por um fluxograma

Mais tarde quando o módulo Leitor de Cartões micro SD foi adicionado à Arquitetura e, consequentemente o programa "MasterV3.ino"foi alterado, adicionando a função "*readFormFile()*", a leitura de ficheiros armazenado no cartão é possível Esta função é chamada quando o "*Smartphone*"envia a mensagem "D"através do Bluetooth e permite realizar a leitura de dados do ficheiro "*DataValues.txt*", situado na raiz do cartão micro SD, enviando essa informação para o "*Smartphone*".

A Figura 4.21 representa o algoritmo desta função por fluxograma.

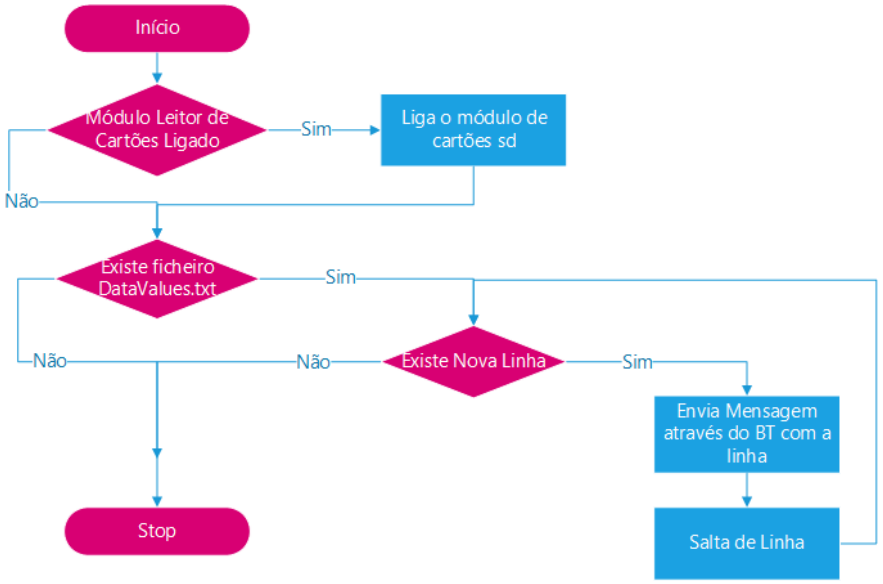

Figura 4.21: Representação da função "readFromFile()"por um fluxograma

4.3.6 MPL115A2

A partir da informação exposta no capítulo "Fundamentos Tecnológicos"na secção "MPL115A2"3.4.3, foi iniciada a integração deste sensor na arquitetura, isto é, integrando o sensor no "*Hop Master*". Sabendo que o sensor MPL115A2 comunica via "I_2C"3.4.3 e qual a descrição de cada terminal descrita na Tabela: 3.15, foi iniciada a sua integração no micro-controlador. A Tabela: 4.13 apresenta a forma como foi conectado os terminais do micro-controlador ao sensor.

Tabela 4.13: Ligação dos terminais do sensor MPL115A2 ao micro-controlador "Arduino"

Terminal do Sensor MPL115A2	**Terminal micro-controlador "*Arduino*"**
VDD	3.3 [V]
GND	GND
SDA	SDA (A4)
SCL	SCL (A5)

A biblioteca "Adafruit_MPL115A2.h"possibilita interligar o sensor MPL115A2 ao micro-controlador "*Arduino*". O objetivo desta biblioteca é descodificar o conjunto de bits enviados pelo protocolo I_2C e retirar os valores de pressão e temperatura.

A partir do código presente nos programas "*getPressure.ino*"e "*getPT.ino*", disponibilizados pela biblioteca, foi verificada a funcionalidade das seguintes funções:

- $Adafruit_MPL115A2()$, construtor de acesso às funções à biblioteca "$Adafruit_MPL115A2$".
- "*void begin(void)*", iniciar o protocolo de comunicação i2c.
- "*float getPressure(void)*", função que devolve a pressão em [kPa].
- "*float getTemperature(void)*, função que devolve a temperatura em [°C].
- "*void getPT(float ∗P, float ∗T)*", função que devolve a pressão em [kPa] na variável P e a temperatura em [°C] na variável T.

Estas funções servem apenas para descodificar os valores de pressão e temperatura em [kPa] e [°C], respetivamente do sensor MPL115A2.

Após estas considerações, também é necessário adicionar a biblioteca "*Wire.h*". Esta biblioteca inicializa a comunicação I_2C, mas também que esta possa operar em sintonia com o resto do hardware.

Com a consolidação dos conhecimentos iniciado a continuação do desenvolvimento do programa "*MasterV3.ino*". Este sensor é usado quando o "*Hop Master*"pretende guardar os novos dados no cartão micro SD provenientes dos "*Hops Slave*". A Figura 4.22 representa o funcionamento em conjunto deste sensor com o módulo leitor de cartões micro SD.

4.3.7 Dispositivos Periféricos

Após a fase de integração dos sensores no "*Hop Slave*", identificou-se dois problemas no "*Hop Master*". O micro-controlador "*Arduino*"não tinha a capacidade de guardar os dados sentidos pelos sensores a longo prazo, adicionalmente é necessário atribuir a data e hora à recolha dos dados. A solução encontrada foi adicionar ao "*Hop Master*"um módulo leitor de cartões micro SD e um "*Real Time Clock*". A integração destes dois periféricos permite aumentar a memória para guardar dados e armazena-los a longo prazo, anexar a data, hora, minuto e segundo aos dados sentidos pelos sensores e resgatar dados do cartão caso a comunicação com o servidor mySQL falhe. Apesar nesta fase do projeto não ser utilizado, todos os dispositivos dentro desta rede "*Mesh*"tem conhecimento de data, hora, minuto e segundo. Em primeiro lugar integrou-se o leitor de cartões SD 3.5.1 e de seguida realizou-se a integração de um "*Real Time Clock*"3.5.2.

A integração do módulo de leitor de cartões tinha terminais em comum com o módulo de comunicação nRF24L01+. Essa sobreposição era imposta pela biblioteca "*SD.h*",

conforme-se no capítulo "Fundamentos Tecnológicos"na secção "Módulo Leitor de Cartões Micro SD"3.5.1. Por forma a evitar a sobreposição de terminais, a solução encontrada foi adicionar a biblioteca "*SDFat.h*"ao projeto. Esta permite flexibilidade nos terminais ao conectar ao micro-controlador "*Arduino*". A Tabela 4.14 demonstra as conexões que foram feitas para integrar este módulo no micro-controlador "*Arduino*", em especifico no "*Hop Master*".

Tabela 4.14: Ligação dos terminais dos sensores aos terminais *Arduino*

Terminal Leitor SD Card	**Terminal Arduino**
5V	5V
GND	GND
MOSI	6
MISO	7
SS	4
SCK	5

Após a integração do leitor de cartões micro SD na Arquitetura, realizou-se a integração do módulo "*Real Time Clock DS1307*"com uma "*EEPROM*", informe-se sobre este componente no capítulo "Fundamentos Tecnológicos"na secção "DS1307 com EEPROM"3.5.2.1. ATabela4.15 apresenta as conexões realizadas que permitiram a interligação do RTC com o micro-controlador "*Arduino*".

Tabela 4.15: Coneção dos terminais dos sensore aos terminais Arduino

Terminal RTC	**Terminal Arduino**
V_{CC1}	5V
GND	"*GND*"
SDA	SDA
SCL	SCL

A integração por software do leitor de cartões micro SD foi realizada no programa "*MasterV3.ino*". Para tal foi adcionada a biblioteca "*SDFat.h*". Como referido anteriormente, a biblioteca "*SD.h*"obrigava que as conexões dos terminais do módulo leitor de cartões microSD fossem as mesmas no módulo de comunicação nRF24L01+. Por essa razão utilizou-se a biblioteca "*SDFat.h*"que permite a liberdade de definir os terminais do módulo leitor de cartões microSD. Para realizar a integração deste módulo no código já existente no programa "*MasterV3.ino*", foi criada a função "*WriteonDataValues()*". Esta função tem a capacidade de avaliar se o módulo de cartões microSD está ligado, e se não estivar, liga o mesmo. Antecipadamente, verifica-se ainda que existe um ficheiro de dados "DataValues.txt"na raiz do cartão. Caso não exista o ficheiro é criado. Após estas verificações, a informação relevante é escrita no ficheiro. A Figura 4.22 representa o algoritmo desta função.

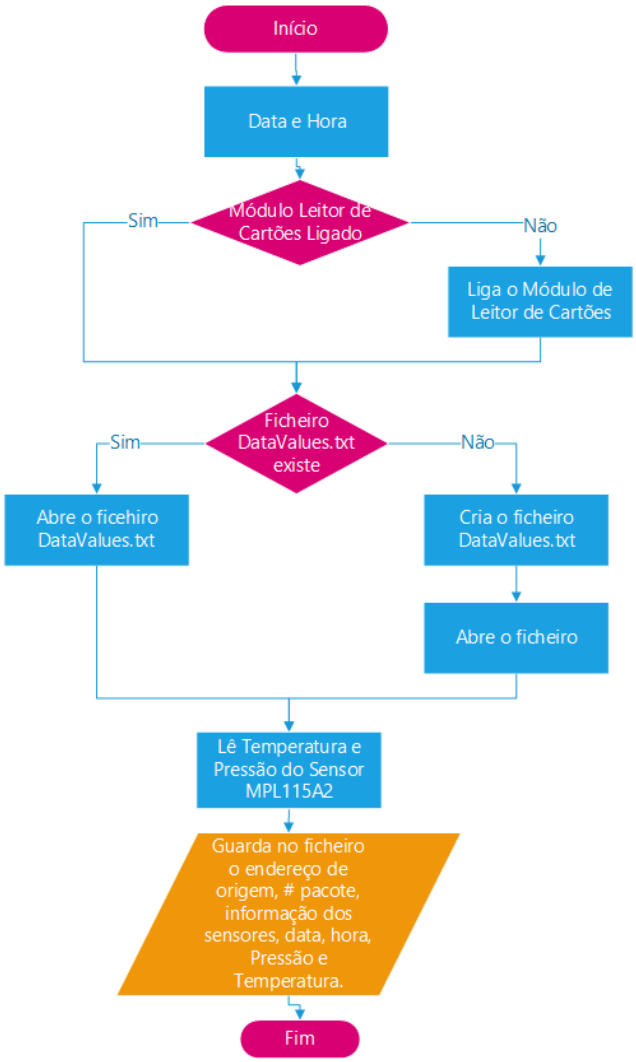

Figura 4.22: Representação da função "WriteOnFiledataValues()"por um fluxograma

Esta função para ser usada corretamente deve ser chamada no processo de receção sempre que exista uma mensagem com o comando S com os dados de sensores.

De seguida foi integrado o módulo RTC. A sua integração no software é simplificada com as funcionalidades presentes na biblioteca "*RTC.h*"pois apesar de reduzidas, permitem a recolha de informação sobre a data e hora.

Foram criadas duas funções, para a recolha de informação relativamente à data e hora e para envio de uma mensagem com a data e hora para os restantes "*Hops*".

A Figura 4.23 é uma representação da função "*getRTCTime()*"e a Figura 4.24 é uma representação da função "*sendRTCTime()*"por fluxograma.

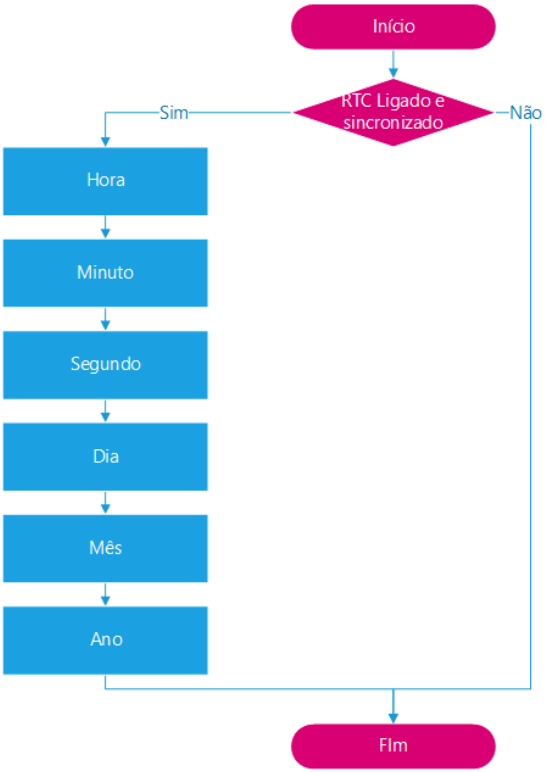

Figura 4.23: Representação da função "*getRTCTime()*"por um fluxograma

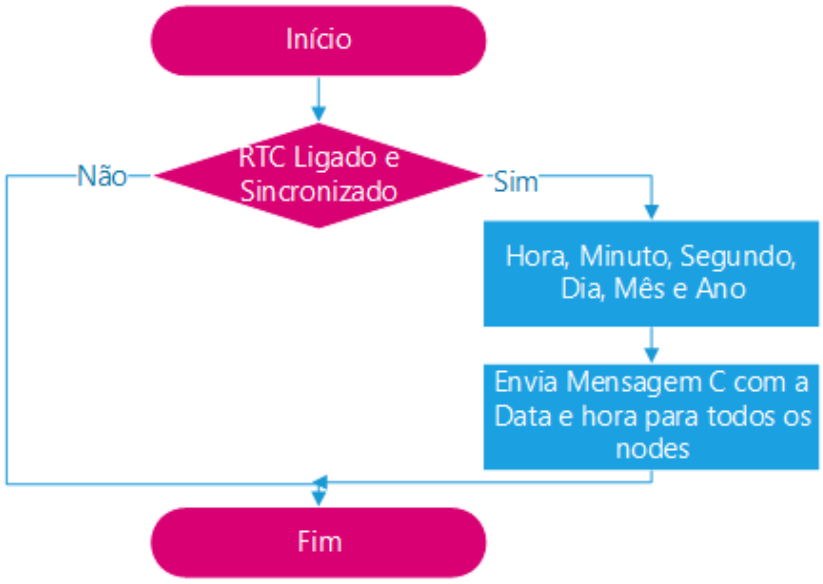

Figura 4.24: Representação da função "*sendRTCTime()*"por um fluxograma

No entanto o módulo "*Real Time Clock*", ao perder a alimentação, necessita de sincronizar a data e hora de novo. Como tal foi criado o programa "*syncDateTime.ino*"que responde a esta necessidade. O programa necessita ainda de ter conectado o cabo usb ao micro-controlador e ao computador, pois a informação a data e hora presente do computador para mais tarde para o módulo RTC. Este último providencia os dados relativos à data e hora ao micro-controlador, quando solicitados, embora este módulo encontra-se constantemente a contabilizar o tempo e a guardar essa informação na EEPROM. Ou seja, o micro-controlador nunca tem a informação da data e hora atualizada exceto quando é efetuado o pedido ao RTC.

A Figura 4.25 é uma representação do algoritmo "*syncDateTime.ino*".

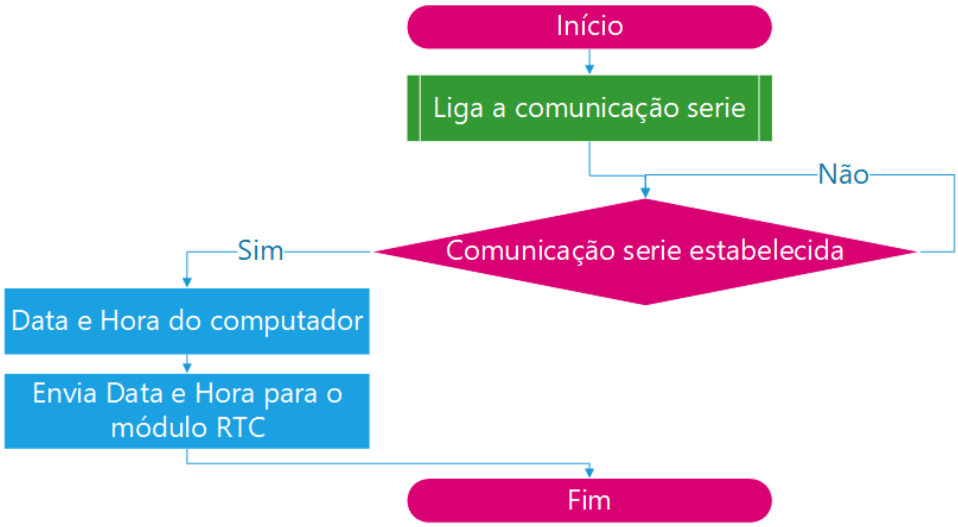

Figura 4.25: Representação do programa "syncDateTime.ino"por um fluxograma

4.3.8 MG811

A integração do sensor de gás MG811 consiste em três fases. Com os conhecimentos adquiridos no capítulo "Fundamentos Tecnológicos"na secção "MG811"3.4.5 e no "*datasheet*"deste sensor [30] foi calculada a função de transferência do gás dióxido de carbono. De seguida foi integrada num micro-controlador "*Arduino*"isolado da Arquitetura. Após os conhecimentos terem sido consolidados nos dois primeiros temas, foi feita a sua integração na Arquitetura num novo "*Hop Slave*"em conjunto com os termopares TMP36 com os conversores A/D Max38155.

4.3.8.1 Calibração e Integração num micro-controlador "Arduino"

O sensor de gás tem como saída valores de tensão como tal, é necessário converter mesmos de tensão obtidos pelo micro-controlador "*Arduino*"em concentração de gás dióxido de carbono, excpresso em [ppm], usando como recurso uma função de transferência. O

primeiro passo foi utilizar a ferramenta *"Excel"*para replicar o gráfico 3.6 para o gás dióxido de carbono. Visto que os gases etanol, metano e monóxido de carbono têm uma característica muito próxima de uma das outras, não faz sentido calcular a função de transferência para estes gases. Os pontos escolhidos deram a origem ao seguinte gráfico expresso na Figura 4.26.

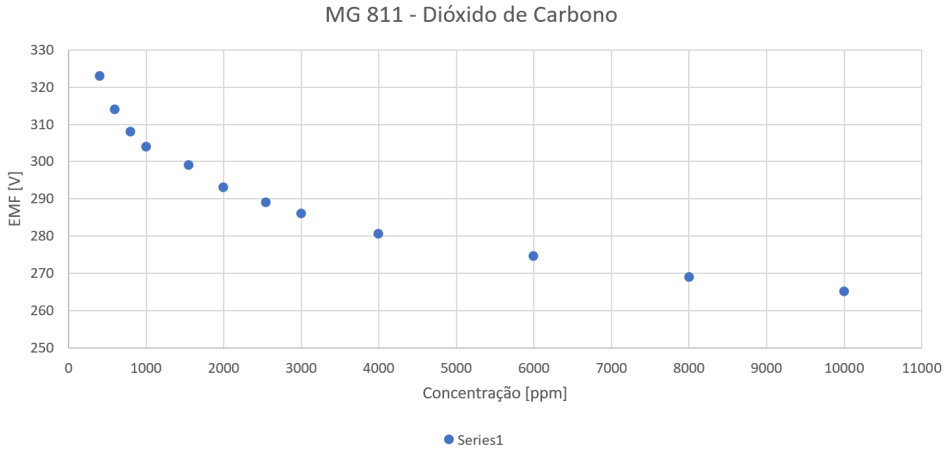

Figura 4.26: Gráfico Dióxido de Carbono de Concentração [ppm] por EMF [V]

Ao observar este gráfico é de notar que o valor de entrada é da concentração do gás dióxido de carbono [ppm] e o valor de saída do mesmo tensão [V].

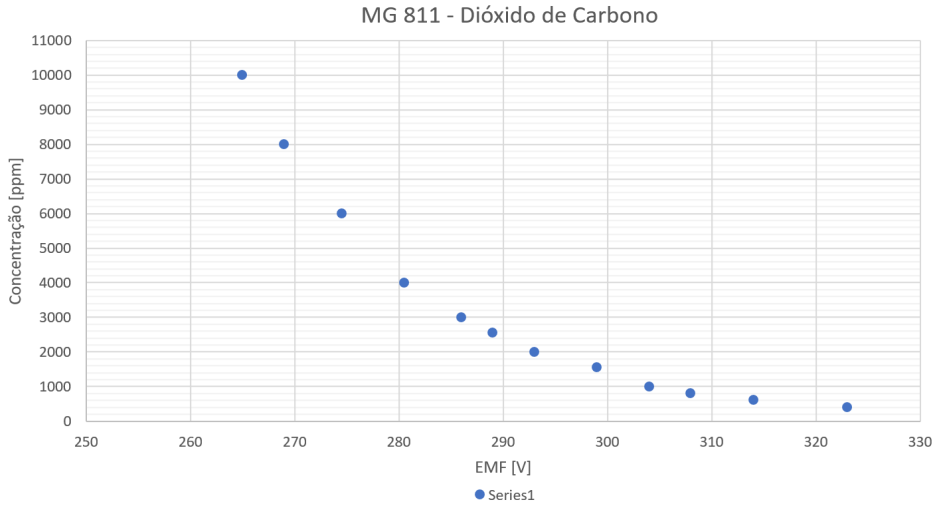

Figura 4.27: Gráfico Dióxido de Carbono de EMF [V] por Concentração [ppm]

Para calcular a função de transferência para do gráfico 4.28 recorreu-se à ferramenta criação de linhas de tendência da aplicação "*Excel*"e tomando como variável a concentração ao invés da tensão.

Podemos observar que neste gráfico que os valores de entrada EMF variam entre 260 a 330 [V] e a saída, a concentração em [ppm] varia entre 0 a 10000 [ppm].

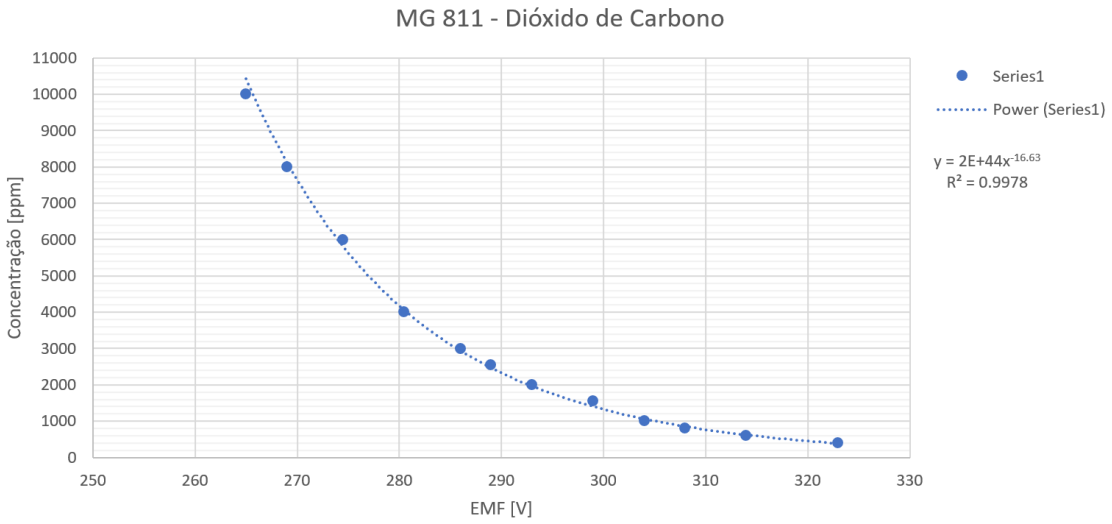

Figura 4.28: Gráfico Dióxido de Carbono de EMF [V] por Concentração [ppm] com a linha de tendência

A linha de tendência utilizada foi a de potência e a função de transferência calculada é expressa na equação 4.13 com o erro quadrático em referência 4.14.

$$ppm = 2 * 10^{44} * EMF^{-16.63} \tag{4.13}$$

$$R^2 = 0.9978 \approx 1 \tag{4.14}$$

Após o cálculo da função de transferência foi criado um programa, "*MG*811_*FT.ino*", através da aplicação "*Arduino IDE*", que permite realizar a leitura dos valores de tensão do sensor MG811 e converte-los para níveis de concentração em [ppm].

O programa "*MG*811_*FT.ino*"permite realizar a conversão do valor sentido pelo sensor e de seguida calcular a concentração do gás dióxido de carbono através da função de transferência 4.13.

A Figura 4.29 é uma representação do algoritmo do programa "MG811_FT.ino".

4.3.8.2 Integração na Arquitetura

Quando os conhecimentos, expostos na secção anterior 4.3.8.1, foram consolidados foi realizada a integração deste sensor na Arquitetura.

Para integrar este sensor na Arquitetura, foi criado um novo "*Hop Slave*"e ao fazê-lo criou-se uma rede de sensores com três "*Hops*". Este "*Hop*"é constituído por um módulo de comunicação nRF24L01+, o sensor MG811 e os termopares TMP36 com o conversor Max38155 que irão ser abordados no tema seguinte 4.3.9.

Iniciou-se esta fase por conectar os terminais deste sensor ao micro-controlador e a Tabela4.16 demonstra como foi feito.

Após as conexões estarem estabelecida, foi iniciado o desenvolvido do programa "*Slave2.ino*". Este programa é baseado no programa "*SlaveV3.ino*", apenas em vez de se

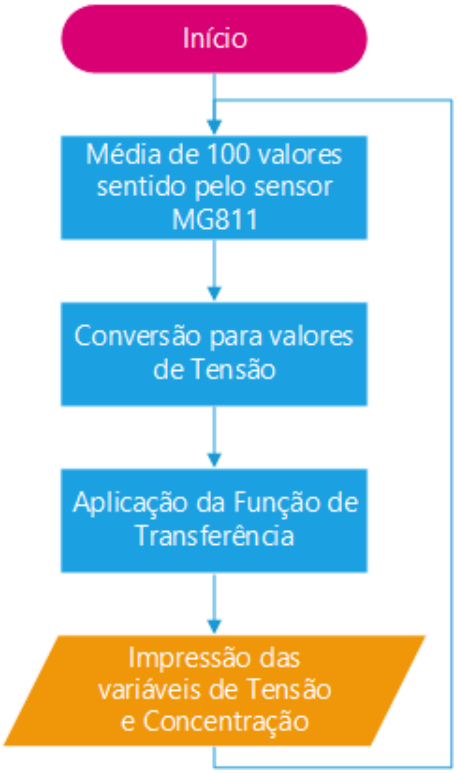

Figura 4.29: Representação do programa "MG811_FT.ino"por um fluxograma

Tabela 4.16: Ligação dos termais do sensor ao micro-controlador "Arduino"

Terminal do sensor MG811	**Terminal do micro-controlador "Arduino"**
V_{CC}	3.3V
GND	GND
A	A0

usarem os sensores Humidade do Solo, MQ2 e DHT11, é utilizado três termopares e o sensor em estudo.

No programa "*Slave2.ino*"a integração deste sensor consiste em criar a função "*MG811ReadCO2()*"no processo "*Timer*".

A Figura 4.30 é uma representação do processo "*Timer*".

A Figura 4.31 é uma representação do algoritmo da função "*MG811ReadCO2()*".

4.3.9 TMP36 e o conversor A/D Max38155

A integração dos componentes termopar TMP36 e o conversor A/D Max38155 foi realizada em duas fases. Os componentes TMP36 e MAX38155 foi montado num micro-controlador "*Arduino*"isolado da Arquitetura. Esta fase é crítica pois permite conhecer o modo de operação destes componentes, bem como a forma como o algoritmo se deve desenvolver para que a integração na Arquitetura seja feita de forma rápida e concisa.

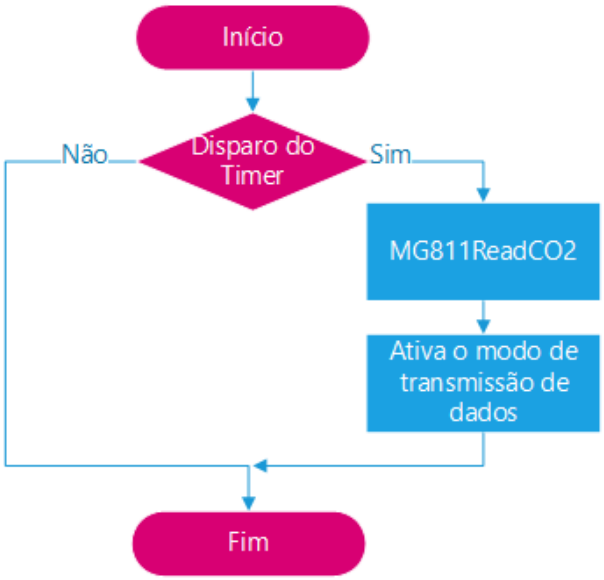

Figura 4.30: Representação da processo de disparo de um Timer por um fluxograma

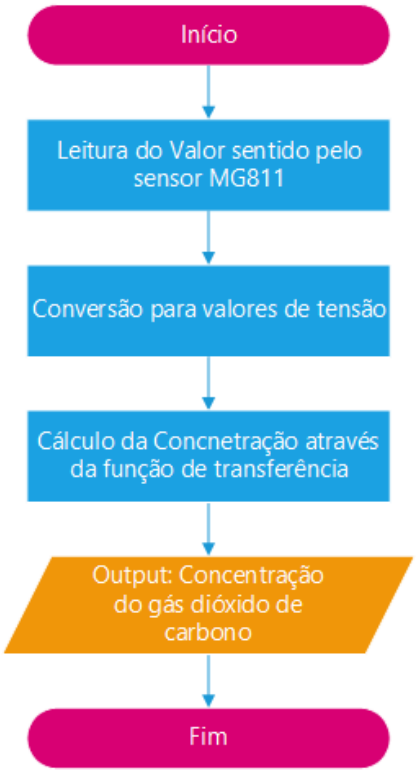

Figura 4.31: Representação da função "MG811ReadCO2"por um fluxograma

Precedeu-se à integração de três termo-pares e de três conversores A/D num micro-controlador "*Arduino*"com um módulo de comunicação nRF24L01+. Esta integração foi realizada em conjunto com o sensor de gás MG811. Ao realizar este passo final criado mais um "*Hop Slave*"a adicionar à Arquitetura.

4.3.9.1 Integração num micro-controlador "Arduino"

A integração no hardware do termopar TMP36 e do conversor A/D Max38155 foi feita em dois passos, foi feita isoladamente pois is mesmos foram adquiridos em separado. Foi necessário conectar os cabos do termopar ao conversor A/D, passo esse foi realizado através da informação exposta no capítulo "Fundamentos Tecnológicos"na secção "Conversor A/D Max38155 com o Termopar TMP36"3.4.1. Após essa ligação ter sido realizada, conduziu-se à ligação dos terminais do conversor ao micro-controlador, conforme a Tabela: 4.17.

Tabela 4.17: ligação dos termais do conversor ao micro-controlador "Arduino"

Terminal do Conversor A/D Max38155	**Terminal do micro-controlador "Arduino"**
V_{in}	-
$3V_o$	3.3V
GND	GND
DO	3
CS	4
CLK	5

Para que os módulos Max38155 e TMP36 operassem em conjunto com o micro-controlador "*Arduino*", foi criado uma biblioteca pela empresa "*Adafruit*", denominada por "Adafruit$_M$AX31855.h". Esta biblioteca permite realizar a conversão dos valores de tensão lida pelo micro-controlador "*Arduino*"para valores de temperatura em [°C] ou e [°F].

Para utilizar esta biblioteca foi ainda necessário incluí-la no código de acordo com o capítulo dos "Fundamentos Tecnológicos"na secção "Programação"na Listagem I.1.

No entanto para que esta biblioteca funcione também é necessário adicionar a biblioteca "*SPI.h*"que permite uso do protocolo "*Serial Protocol Interface*". Foram variáveis com as portas digitais atribuídas no micro-controlador "*Arduino*", declaradas como variáveis globais, de acordo com a listagem 4.1.

Listagem 4.1: Variáveis Globais

```
1  #define MAXDO   3
2  #define MAXCS   4
3  #define MAXCLK  5
```

As funções disponíveis na biblioteca, que permitem a utilização deste sensor são:

- Adafruit_MAX38155 thermocouple(MAXCLK, MAXCS, MAXDO): cria um construtor com as portas digitais. Este construtor permite o uso das funções presentes na biblioteca "Adafruit$_{M}$AX31855.h".
- "*thermocouple.readInternal()*": devolve a temperatura interna do conversor A/D max38155.
- "*thermocouple.readCelsius()*": devolve a temperatura em [°C] do TMP36.
- "*thermocouple.readFarenheit()*": devolve a temperatura em [°F] do TMP36.
- "*thermocouple.readError()*": verifica erros no protocolo SPI.

A biblioteca dispõe de dois programas de teste. São eles o "lcdthermocouple.ino"e "*serialthermocouple.ino*". O primeiro permite realizar a integração de um ecrã lcd e disponibiliza a informação do conversor A/D Max38155 no mesmo. O segundo demonstra como recolher a variável temperatura do conversor e imprime a variável temperatura no "*Serial Monitor*".

4.3.9.2 Integração na Arquitetura"

A integração destes dois componentes na Arquitetura deu-se em conjunto com o sensor de gás MG811. Esta integração implicou a criação de um novo "*Hop Slave*", pois este "*Hop*"tem o objetivo de monitorizar variáveis de interesse para a casa com a tecnologia de lama. A criação de um novo "*Hop Slave*"implicou a integração do módulo nRF24L01+, como descrito na secção 4.3.1.1 no §1.

Após a verificação do correto funcionamento do módulo de comunicação nRF24L01+, através dos conhecimentos adquiridos na secção 4.3.9.1, foi realizada a integração de três termopares no novo "*Hop Slave*". Esta integração implicou o desenvolvimento do programa "*Slave2.ino*", no entanto os conhecimentos adquiridos que deram resultado ao programa "*SlaveV3.ino*"foram utilizados para o desenvolvimento do novo programa. A representação geral do novo programa é exposto pelo fluxograma na Figura I.17. Este programa, à semelhança do programa desenvolvido "*SlaveV3.ino*", contínua a ter um processo de comunicações, de "*Timer*", de receção e de transmissão de mensagens. Deste modo, única alteração que foi realizada no novo programa foi no processo de "*Timer*". A Figura 4.32 é a representação do processo de "*Timer*"por fluxograma.

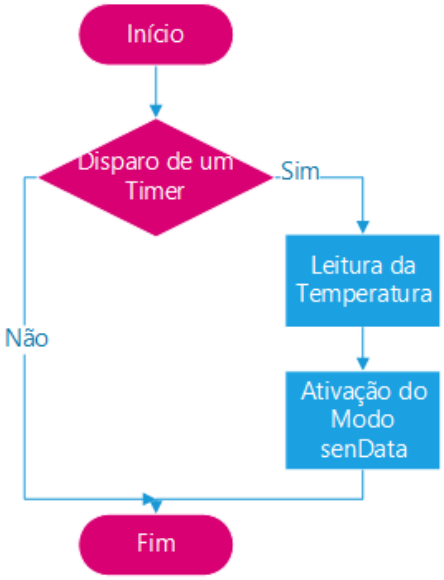

Figura 4.32: Representação do processo "*Timer*"do programa "*Slave2.ino*"por um fluxograma

4.3.10 Esquemáticos *Hop Master* e *Hops Slave*

4.3.10.1 Hop Master

O "*Hop Master*"é constituído pelos micro-controlador "*Arduino*", módulo de comunicação nRF24L01+, módulo de comunicação "*Bluetooth*"BT HC-06, sensor de temperatura e pressão atmosférica MPL115A2, módulo "*Real Time Clock*"DS1307 com uma EEPROM e um módulo leitor de cartões micro SD. Este "*Hop*"tem a capacidade de guardar os dados proveniente de outros "*Hops*", enviar essa informação para um "*smartphone*"e de sentir as variáveis de Temperatura [°C] e a pressão atmosférica em [KPa].

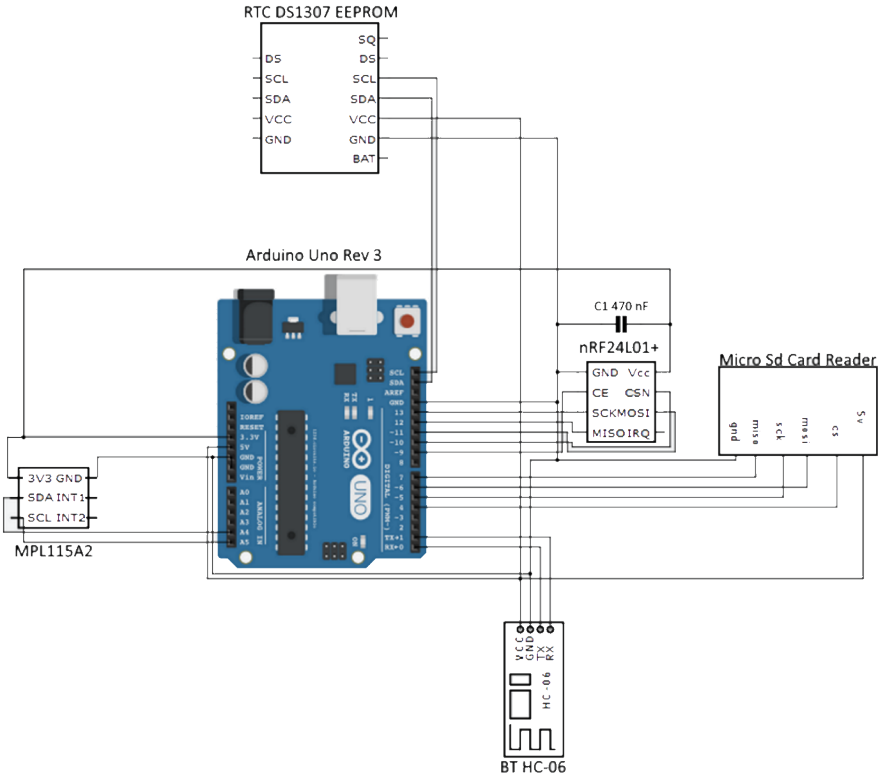

Figura 4.33: Esquemático do Hop Master

4.3.10.2 Hop Slave1

O "*Hop Slave 1*"é constituído pelos micro-controlador "*Arduino*", módulo de comunicação nRF24L01+, sensor MQ2, pelo sensor Humidade do Solo e o sensor DHT11. Este "*Hop*"tem a capacidade de monitorizar as variáveis de Temperatura do ar [°C], a humidade do ar [%RH], a concentração dos gases do Hidrogénio (H_2), Gás de petróleo liquefeito (GPL), Metano (CH_4), Álcool e Propano. Com o sensor de Humidade do solo apenas realiza um cálculo de uma estimativa de massa de água presente num determinada massa de solo. Após a receção dos dados sentidos pelos sensores, esta informação é enviada para o "*Hop Master*".

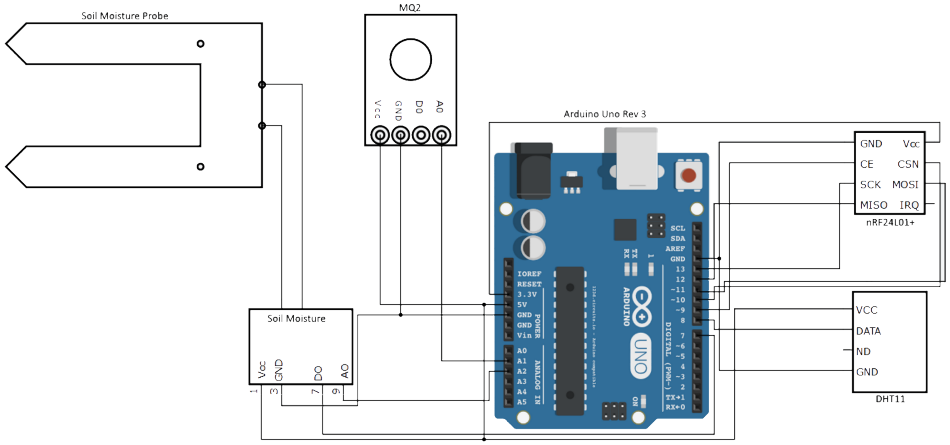

Figura 4.34: Esquemático do Hop Slave 1

4.3.10.3 Hop Slave2

O "*Hop Slave 2*"é constituído pelos micro-controlador "*Arduino*", módulo de comunicação nRF24L01+, o sensor MG811 e três conversores A/D Max38155 com os termopares TMP36. Este "*Hop*"tem a capacidade de sentir a Temperatura [°C] [°F] e a capacidade de calcular a concentração do gás dióxido de carbono (CO_2). Após a informação sentida pelos sensores ser processada, esta é enviada para o "*Hop Master*".

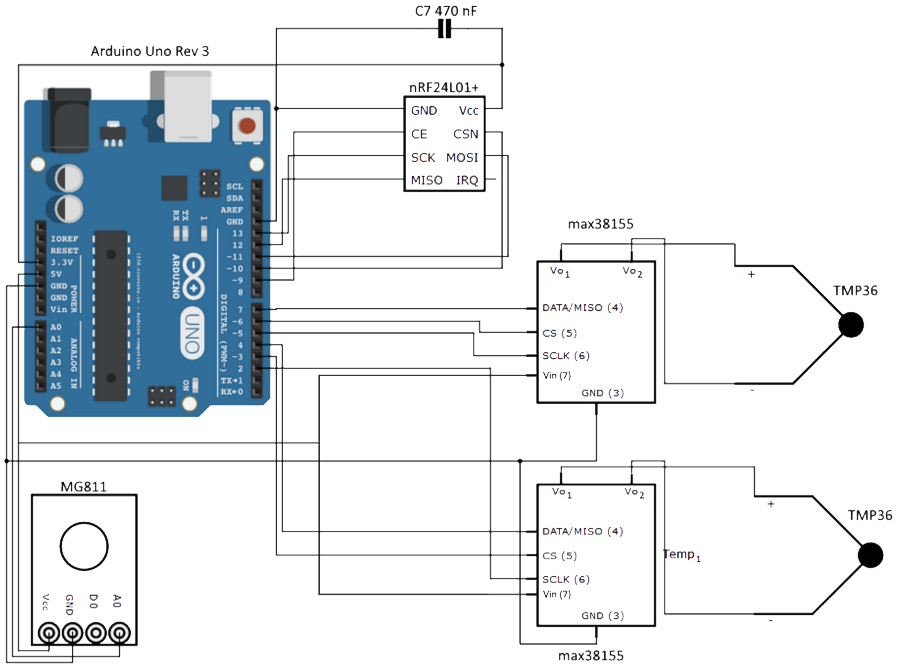

Figura 4.35: Esquemático do Hop Slave 2

4.4 Segunda fase: Servidor de dados com uma interface gráfica

O objetivo de criar um servidor com uma base de dados é permitir que os dados sentidos pelo sensores sejam guardados no servidor. A interface gráfica é apenas um sítio que permite a visualização do conteúdo em formato gráfico, isto é, é gerado nas páginas web informação gráfica que o utilizador poderá visualizar.

Uma das grandes vantagens da utilização de um sítio é que este pode ser consultado em qualquer local, desde que exista uma ligação à Internet estável. Porém, ao implementar um servidor HTTP é necessário abrir as portas HTTP correspondentes no "*router*", o que levanta questões de segurança na rede. A solução encontrada foi implementar uma "*Virtual Private Network*"(VPN) e o protocolo utilizado foi o "*Point-to-Point Tunnelling*"(P2P), de acordo no capítulo "Fundamentos Tecnológicos"sobre estes dois temas 3.8 3.8.2 ,respetivamente.

Todavia para que o sítio esteja disponível para consulta, foi instalado um servidor que compila as linguagens de programação web HTML, CSS, PHP, JavaScript e uma base de dados em MySQL.

4.4.1 Instalação e Configuração do Sistema Operativo

Para implementar os passos anteriormente referidos, em primeiro lugar foi necessário escolher e instalar o sistema operativo para o mini-pc "*Raspberry Pi*".

O site do mini-pc [40] oferece duas hipóteses. Uma delas é o sistema "*Noobs*", que é um sistema operativo mais leve, no entanto com a instalação dos pacotes necessários para o funcionamento do servidor acabaria por ficar com o mesmo nível de processamento do sistema operativo "*Raspbian*". Ambos são sistemas operativos "*Linux*"baseados na distribuição "*Debian*". Pelas razões apresentadas anteriormente optou-se pelo sistema operativo foi o "*Raspbian*"3.7.2.

Para instalar o sistema operativo "*Raspbian*"foi necessário um cartão SD e usar o software "Win32 Disk Image"que permite criar uma instalação do sistema operativo no cartão [56].

Ao configurar o sistema operativo foi garantida a segurança e o bom funcionamento da máquina. Adicionalmente garantiu-se ainda uma ligação de trabalho remoto. Os pontos seguintes demonstram como foi elaborado:

1. Entrar na máquina com o usuário: "*pi*"e a palavra de acesso: "*raspberry*", que são as credenciais de acesso por defeito dos sistemas operativos para as máquinas "*Raspberry Pi*";

2. Abrir o menu de configuração - "*sudo raspi-config*";

3. Alterar a palavra-chave de acesso escolhendo o menu "*Change User Password*";

4. Inserir uma nova palavra-chave de acesso;

5. Permitir o protocolo de acesso "*ssh*"ou "*telnet*"escolhendo o menu "*Advanced Options*"e de seguida "*Enbable SSH Protocol*";

6. Alterar o teclado e o local no menu "*Internationalisation Options*";

7. Sair do menu de configurações selecionando o botão "*<finnish>*";

8. Atualizar o sistema operativo - "*sudo apt-get update && sudo apt-get upgrade -y*";

9. Reiniciar o sistema para que todas as alterações anteriormente feitas sejam aplicadas.

4.4.2 Putty

Uma ligação remota foi criada para que o processo de instalação e configuração da VPN do servidor fosse facilitada. O protocolo SSH foi iniciado no mini-pc para permitir ligações de trabalho remota.

Através do programa "*Putty*", em sistema operativo Windows, foi estabelecida a estação de trabalho remota através da mesma rede.

4.4.3 Instalação e Configuração da Virtual Private Network Point-to-Point Tunelling

Através dos conhecimentos adquiridos no capítulo "Fundamentos Tecnológicos"na secção "Virtual Private Network"3.8, a VPN permite aceder a máquinas dentro de outras redes. O processo de instalação da VPN em Linux é é efectuado através do seguinte comando - "*sudo aptitude install pptpd -y*".

Para configurar a VPN é necessário editar três ficheiros do pacote "*pptpd*"e ainda adicionar uma "*firewall*"3.7.3 no mini-pc, garantindo alguma segurança nas ligações. Para configurar corretamente a "*VPN*"é necessário executar os seguintes pontos:

1. Editar o ficheiro que se encontra na diretoria "*/etc/pptpd.conf* - "*sudo nano /etc/pptpd.conf*";

2. Adicionar no fim do ficheiro as seguintes linhas de código de acordo com a listagem 4.2;

Listagem 4.2: Edição do ficheiro pptpd.conf

```
1       localip: 192.168.1.101
2       remoteip 192.168.1.102-254
```

Onde a primeira linha de código indica o IP que o mini-pc "Raspberry Pi"tem e a segunda linha significa os IP disponíveis para novas ligações. Através do exemplo mencionado na listagem 4.2 existem 152 IP disponíveis para atribuir a novas ligações;

3. Editar o ficheiro que se encontra na diretoria "*/etc/ppp/pptpd-options*- "*sudo nano /etc/ppp/pptpd-options*";

4. Adicionar no fim do ficheiro as seguintes linhas de código de acordo com a listagem 4.3;

Listagem 4.3: Edição do ficheiro pptpd-options

```
1       nobsdcomp
2       noipx
3       mtu 1490
4       mru 1490
```

5. Editar o ficheiro na diretoria "*/etc/ppp/chap-secrets*", permitindo adicionar novos utilizadores à VPN - "*sudo nano /etc/ppp/chap-secrets*". O utilizador e a palavra chave são adicionados de acordo com a listagem reflst:VPN-chap-secrets.

Listagem 4.4: Edição do ficheiro chap-secrets

```
1       usuário 1<TAB>*palvra-chave de acesso<TAB>*
```

6. Por fim, para que as edições sejam aplicadas, é necessário reiniciar o serviço "*pptpd*- - "*sudo /etc/init.d/pptpd restart*".

Do ponto de vista de segurança informática faz sentido criar uma "*firewall*"para limitar o tipo de acessos ao mini-pc e os protocolos de comunicação usados.

1. Criar um ficheiro "*script*- "*sudo nano /etc/init.d/firewall.sh*";

2. Adicionar o código de acordo com a listagem 4.5;

Listagem 4.5: Criação de um ficheiro firewall.sh

```
1       !/bin/bash
2       iptables -I INPUT -p tcp --dport 1723 -m --state NEW -j ACCEPT
3       iptables -I INPUT -p gre -j ACCEPT
4       iptables -t nat -I POSTROUTING -o etho -j MASQUERADE
5       iptables -I FORWARD -p tcp --tcp-flags SYN,RST SYN -s
6       192.168.1.102-254 -j TCPMSS --clamp-mss-to-pmtu
```

3. Criar um executável para o ficheiro "*firewall.sh*- "*chmod 700 /etc/init.d/firewall.sh*";

4. Executar o novo executável - "*chown root /etc/init.d/firewall.sh*";

5. Colocar o novo executável na lista de programas que arrancam após o "*boot*"da máquina. Neste caso, este executável irá ser executado três segundos depois do arranque do "*boot*- "*update-rc.d /etc/init.d/firewall.sh defaults 100*".

4.4.4 Instalação do Servidor

Como já foi referido no início desta secção 4.4, é necessário criar um servidor que seja capaz correr ficheiros com linguagens de programação HTML, PHP, JavaScript. o software Apache2 3.9 é perfeitamente capaz de responder aos requisitos das linguagens de programação usadas. No entanto, é necessário criar uma base de dados em "*MySQL*"e o software utilizado foi o "*mysql server*". Para o ambiente gráfico mais técnico de acesso às bases de dados foi utilizado o software "*php my admin*". Os próximos passos descrevem como foi feita a instalação do software:

1. Instalação do software my Sql server - "*sudo apt-get install mysql-server*";
2. Inserir uma palavra chave de acesso à base de dados;
3. Instalação do "*Apache 2*"e do "*PHP5*- "*sudo apt-get install apache2 php5 libapache2-mod-php5*";
4. Instalação do "*php my admin*- "*sudo apt-get install phpmyadmin*";
5. Inserir a palavra-chave de acesso às bases de dados.

4.4.5 Configuração do Apache 2

Para que o "*PHP My Admin*"entre em funcionamento quando o "*Apache 2* é iniciado é necessário editar um ficheiro de configurações do software.

1. Edição do ficheiro na diretoria "*/etc/init.d/apache2/apache2.conf*- "*sudo nano /etc/init.d/apache2/apache2.conf*";
2. Adicionar a seguinte linha de código no fim do ficheiro expressa na listagem 4.6;

Listagem 4.6: Configuração apache2.conf

```
1       include /etc/phpmyadmin/apache.conf
```

3. Iniciar o serviço para que as alterações sejam aplicadas - "*sudo /etc/init.d/apache2 restart*".

Os novos ficheiros do site são guardados na diretoria "*/var/www/html*", e se não o forem guardados nesta diretoria o software não os reconhece.

Para aceder ao sítio, é necessário abrir uma página no "*browser*"e colocar no campo URL o seguinte "*192.168.1.101/site*". Com este passo se o servidor "*Apache2"* estiver a correr deverá mostrar o site.

4.4.6 Bases de dados

A base de dados serve apenas como repositório dos dados sentidos pelos sensores provenientes do dispositivos "*Smartphone*". Esta base de dados é composta por três tabelas. A Tabela"*Users*", que tem a capacidade de guardar um utilizador com uma palavra passe e um email, tem dois propósitos. Um primeiro é relacionar um dispositivo "*Hop*"a um utilizador e um segundo serve apenas para fazer verificação de "*LogIn*". A Tabela "*nodes*"apenas serve para guardar a informação de quantos sensores tem cada "*Hop*"e se estes são "*Master*"ou não. Finalmente a Tabela mais importante é a que permite cruzar a informação dos sensores com a descrição de cada "*Hop*"e utilizador.

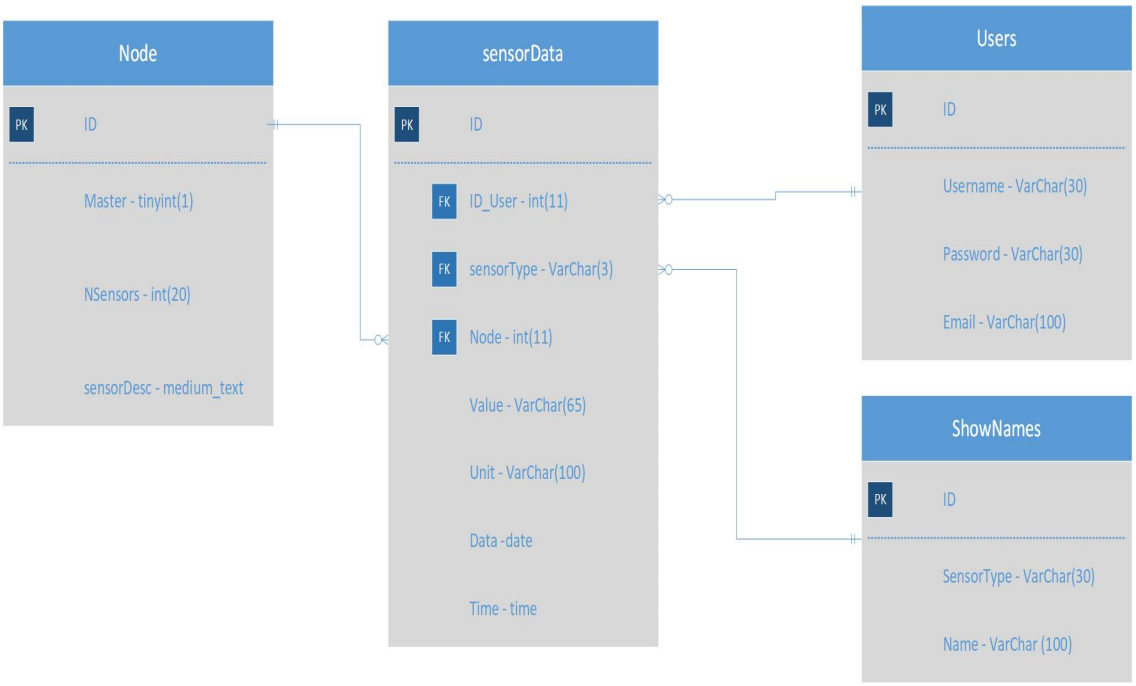

Figura 4.36: Diagrama de relação e de entidade da base de dados

4.4.7 Sítio Web Desenvolvido

A base de dados, referenciada na secção anterior 4.4.6 apenas permite a salvaguarda da informação, o que por si só não permite a visualização fácil dos dados. Porém o sitio web desenvolvido tem a capacidade de gerar gráficos a partir da informação retida na base de dados.

Para desenvolver o sítio foi usado um "*template*"para facilitar o processo deste desenvolvimento. Consulte o sítio [50]. A geração dos gráficos é realizada através do pacote HighCharts, consulte o sítio [23].

4.5 Terceira Fase: Desenvolvimento de uma aplicação para *Smartphone*

A aplicação tem o objetivo de realizar a ponte entre os dispositivos "*Hops*"com os vários sensores e uma base de dados, local onde fica armazenada a informação sentida pelos mesmo.

Esta aplicação foi desenvolvida tendo em consideração os seguintes requisitos realizar uma ligação ao "*Hop Master*"através da tecnologia "*Bluetooth*", guardar os dados no "*Smartphone*", quando existir uma ligação estável à "*Internet*"enviar os dados para um servidor com uma base de dados.

Inicialmente, o desenvolvimento da aplicação tinha sido feita na plataforma "*Xamarin*"para o IDE "*Visual Studio*". Esta plataforma foi escolhida pois permite o desenvolvimento de aplicações para os sistemas operativos "*Android*", "*IOS*"e "*Windows Phone*". No entanto, devido ao facto de possuir apenas um "*Smartphone*"com o sistema operativo "*Android*", a aplicação apenas será testada neste sistema operativo. As outras razões que levaram a escolha desta plataforma é a linguagem C# e por operar no "*IDE Visual Studio*".

Todavia, não foi possível concluir o desenvolvimento da aplicação nesta plataforma, pois um dos requisitos mais importantes não estava a ser cumprido. A data limite para a entrega desta dissertação, o incumprimento do requisito de guardar os dados localmente no "*Smartphone*", a ligação ao servidor ainda não estar implementada e a aprendizagem lenta, foram os fatores que levaram a optar o desenvolvimento da aplicação em outra plataforma.

A plataforma escolhida foi "MIT APP INVENTOR 2". As razões que levaram à escolha desta plataforma são a fácil utilização, rapidez na aprendizagem do uso das ferramentas disponíveis e interface intuitiva. Com esta plataforma foi possível desenvolver a aplicação, em tempo útil, respeitando todos os requisitos inicialmente impostos.

O funcionamento da aplicação, BeeNodes, está representado pelo fluxograma na Figura 4.37.

Esta aplicação tem cinco funcionalidades que podem ser executadas em paralelo.

1. A funcionalidade à direita verifica se o dispositivo "*Bluetooth*"está ligado, caso este não esteja ativo imprime a mensagem "*Bluetooth is not Connected*".

2. A segunda funcionalidade à direita, é uma lista, ao ser pressionada procura os dispositivos "*Bluetooth*"emparelhados. Após a lista estar completa pode-se escolher o dispositivo. É nesta fase em que o "*Smartphone*"se liga ao "*Hop Master*".

3. Esta funcionalidade, é um botão, realiza o pedido de dados ao "*HopMaster*". Posteriormente, este último envia a informação que será guardada num ficheiro.

4. A funcionalidade seguinte, é um botão que envia a informação presente no ficheiro para o servidor, utilizando o método Get.

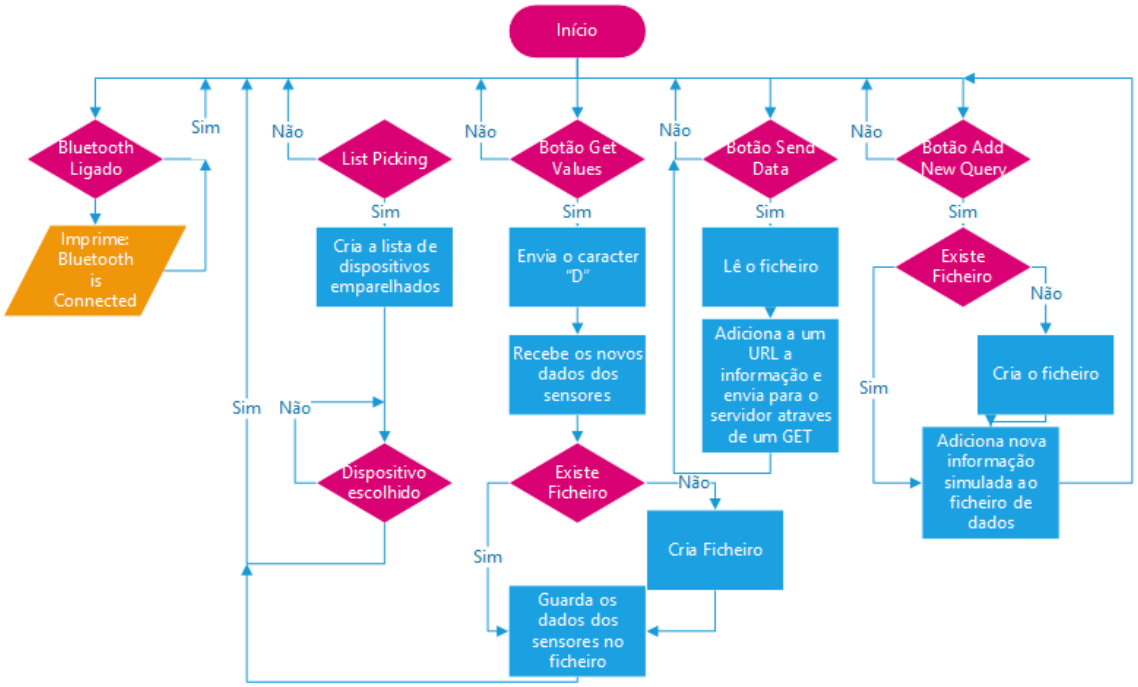

Figura 4.37: Fluxograma da aplicação BeeNodes

5. Esta funcionalidade, é um botão, simula um dado que posteriormente pode ser enviado para o servidor.

O diagrama de sequência 4.38 indica as comunicações realizadas entre a aplicação e o "*Hop Master*", e entre a aplicação e o "*Smartphone*".

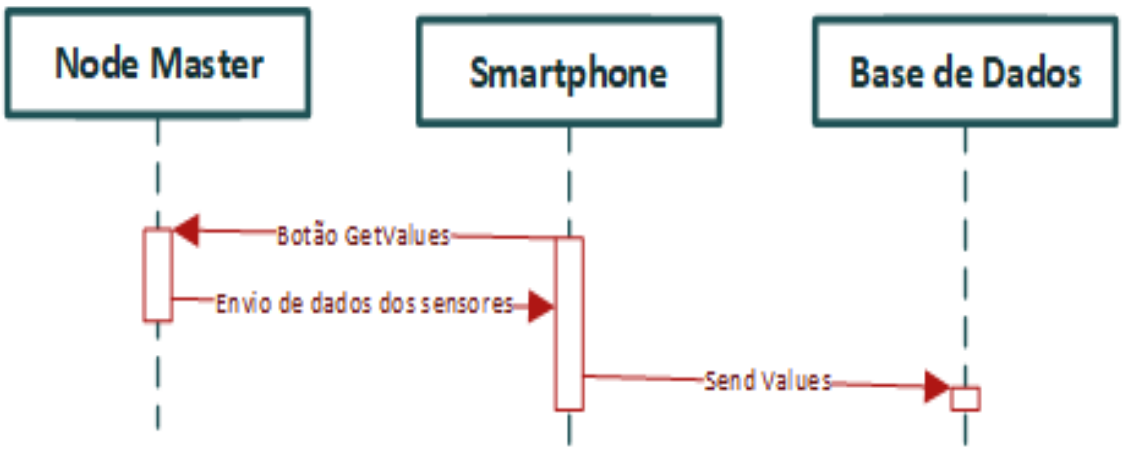

Figura 4.38: Diagrama de sequência para as comunicações efetuadas pela aplicação BeeNodes

CAPÍTULO 5

CONCLUSÃO

Na fase de estudo com o micro-controlador "*Pinoccio*"foi entendido que este era capaz de criar redes usando o protocolos WAP e WEP no entanto, este micro-controlador não era capaz de permanecer ligado quando estas redes eram criadas. Ou seja, o micro-controlador criava a rede e esta permanecia ligada durante 5 minutos, desligando-se após esse tempo cessar.

A plataforma "*Pinoccio*", para o browser "*Google Chrome*", apenas funcionava em máquinas "*Mac OS*". No entanto, os sistemas operativos "*Linux*"e "*Windows*"também foram testados, embora sem sucesso. Para não cessar este assunto, a plataforma também foi testada em uma máquina virtual "*Mac OS*"num sistema operativo "*Windows*", porém nem sempre a integração do micro-controlador com a plataforma era realizada com sucesso.

Para simplificar o carregamento do software para o hardware, foi adicionado um botão de "*reset*". Tal era necessário porque o programa base era muito extenso e o micro-controlador não era capaz de sincronizar a comunicação serie com o computador, o que tornava o carregamento de software impossível. Este botão servia para desligar o micro-controlador e no instante certo liberta-se para que o software pudesse ser carregado.

Estes foram os motivos pelos quais levou criar uma nova solução para o problema e desenvolvendo-se assim a arquitetura atual.

Uma das vantagens da utilização de uma rede de sensores "*Mesh*"é poder integrar, com alguma liberdade, os sensores necessários na arquitetura e registar os dados sentidos noutro "*Hop*". Criando assim alguma segurança no registo dos dados sentidos.

Como os "*Hops*"criados apenas usam um módulo de comunicação nRF24L01+, implica que a geração "*Mesh*"implementada foi a de primeira geração, porém ao adicionar mais um rádio, a rede poderá evoluir para a segunda geração. Ao trocar um dos módulos de comunicação, da segunda geração, por um módulo capaz de criar uma rede a 5 [GHz] é possível implementar a terceira geração de redes "*Mesh*".

Com a criação de um servidor capaz de albergar uma base de dados e um sítio web, é possível tirar algum processamento da rede de sensores e aplica-lo no servidor, por exemplo, as funções de transferência que permitem converter dados sentidos pelos sensores em variáveis podem ser calculadas no servidor e apresentar os dados corretos no sítio web. Para além deste pormenor, também é possível criar outros serviços no servidor, como por exemplo, notificações "*push*".

A aplicação "*Smartphone*"inicialmente foi desenvolvida na plataforma "*Xamarin*". Como o prazo de entrega da dissertação estava muito próximo, obstáculos ainda por ultrapassar e os requisitos ainda por cumprir levaram a mudar de plataforma. A plataforma escolhida foi a "MIT App Inventor 2", que possibilitou o desenvolvimento da aplicação com sucesso e em tempo útil. Ou seja todos os requisitos inicialmente impostos foram cumpridos com sucesso.

No fim do desenvolvimento da arquitetura esta foi testada com sucesso. Todavia, o micro-controlador usado não é o melhor a aplicar para esta dimensão de projeto. O poder de processamento mo micro-controlador aplicado é limitado para as exigências deste projeto. A sua memória limitada fez com que o programa "MasterV3.ino"fosse limitado no numero de tarefas a executar.

As tarefas que podem ser implementadas num micro-controlador com mais capacidade são: calibrar a data e hora do módulo RTC através da ligação Bluetooth e permitir também ligações Bluetooth Low Energy através do módulo nRF24L01+.

O módulo nRF24L01+ tiveram uma performance extraordinária, a rapidez de comunicação é suficiente para transmitir num curto período de tempo várias mensagens. Apesar deste módulo ter sido usado para criar uma rede de sensores "*Mesh*"este também é capaz de criar redes WPA, WEP e Bluetooth Low Energy.

As bases de dados que foram implementadas no mini-pc "Raspberry Pi", encontram-se em um cartão SD. Estas bases de dados tiveram de ser otimizadas pois o espaço em memória é limitado, porém o grande volume de dados na máquina, não apresentou fracos desempenhos.

O Hop Master, a aplicação BeeNodes e o servidor HTTP com a base de dados, são locais onde fica registada a informação sentida pelos sensores. No caso de se perder informação durante as comunicações, esta pode ser recuperada em um dos outros locais.

No futuro aconselho em projetos de desenvolvimento de hardware, antes de desenvolver o software final, construir uma PCB com todos os componentes de interesse. Realizando este passo evita problemas de maus contactos, a danificação de módulos e fios partidos.

Os departamentos de Engenharia Civil e o Engenharia do Ambiente podem retirar uso deste projeto, pois este permite a realização de monitorização de variáveis de ambiente em espaços fechados. Ou seja o Departamento de Engenharia Civil pode usar este projeto para avaliar ambientes perigosos e o Departamento de Engenharia do Ambiente pode utilizar este projeto para a monitorização de diversas variáveis de ambiente.

Os resultados apresentados indicam que o sistema funciona bem, foi otimizado para a monitorização das variáveis de interesse e garante a consistência dos dados que são os objetivos principais desta dissertação. Com esta nova capacidade os projetos que poderão vir utilizar este projeto estão nas condições de monitorizar variáveis de interesse e analisar os dados obtidos.

Esta arquitetura pode ser aplicada em várias situações, como por exemplo: - Monitorizar o efeito térmico de uma casa contemporânea. - Monitorizar o consumo energético de eletrodomésticos de uma casa. - Monitorizar as luzes ligadas de uma casa durante o período da noite para estimar o consumo energético de uma divisão. - Monitorizar os níveis de monóxido de carbono em locais previamente selecionados de uma oficina. - Monitorizar a frequência cardíaca dos utentes de um ginásio. Monitorizar a quantidade de produtos existentes numa prateleira numa loja.

O trabalho futuro pode passar pelo desenvolvimento da 2ª e 3ª geração "*Mesh*", a implementação de outros sensores fora do domínio de variáveis de ambiente, implementação de controladores, reduzir a escala do hardware e utilizar outros micro-controladores para estudar desempenhos diferentes.

O percurso percorrido durante o desenvolvimento da dissertação, os resultados finais, os obstáculos por ultrapassar, os obstáculos ultrapassados, os requisitos impostos pelo professor orientador, os requisitos pessoais impostos e os requisitos cumpridos são fatores quando avaliados representam que o percurso percorrido até à entrega desta dissertação foi positiva.

BIBLIOGRAFIA

[1] l. Ada. *DHTxx Sensors*. 2016.

[2] T Ahonen, R Virrankoski e M Elmusrati. "Greenhouse Monitoring with Wireless Sensor Network". Em: *Mechtronic and Embedded Systems and Applications, 2008. MESA 2008. IEEE/ASME International Conference on* (2008), pp. 403–408.

[3] Arduino. *Arduino Homepage*. 2016. URL: https://www.arduino.cc/ (acedido em 29/07/2017).

[4] *Arduino - NetBeans Plugin*. URL: http://plugins.netbeans.org/plugin/46054/arduino (acedido em 01/08/2017).

[5] *Arduino IDE*. URL: https://www.arduino.cc/en/Main/Software (acedido em 01/08/2017).

[6] *Arduino IDE for Visual Studio*. URL: https://marketplace.visualstudio.com/items?itemName=VisualMicro.ArduinoIDEforVisualStudio (acedido em 01/08/2017).

[7] *Arduino References*. URL: https://www.arduino.cc/en/Reference/HomePage.

[8] *Arduino Serial*. URL: https://www.arduino.cc/en/Reference/Serial.

[9] *Arduino Uno Rev3*. URL: https://store.arduino.cc/arduino-uno-rev3 (acedido em 21/07/2017).

[10] *Argus HOMEPAGE*. URL: http://www.arguscontrols.com/system/ (acedido em 24/02/2017).

[11] "Atmel AVR 8-and 32-bit Microcontrollers AVR910: In-System Programming". Em: (2016).

[12] *Autodesk Circuits*. URL: https://circuits.io/ (acedido em 01/08/2017).

[13] S. Bazaar. "Grove -Moisture Sensor". Em: *Seeed Technology Inc.* (2015), pp. 1–13.

[14] "Bluetooth to serial HC-06 wireless module Product". Em: ().

[15] "Climate Control Systems INC Homepage". Em: (). URL: http://www.climatecontrol.com/ (acedido em 24/02/2017).

[16] V. R. Deore e P. V. M. Umale. "Wireless Monitoring of the Green House System Using Embedded Controllers". Em: 3.2 (2012), pp. 1–8.

[17] *Download Raspbian for Raspberry Pi*. URL: https://www.raspberrypi.org/downloads/raspbian/ (acedido em 07/09/2017).

[18] J. Fang e F. Wang. "Design of Greenhouse remote monitoring system based on LabVIEW". Em: *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on* 1 (2011), pp. 536–539. DOI: 10.1109/CSAE.2011.5953277.

[19] J. M. Fonseca. *Slides das Aulas da cadeira Sistemas Sensoriais*.

[20] *Gravimetric & Volumetric Soil Water Content | Edaphic Scientific*. URL: http://www.edaphic.com.au/soil-water-compendium/how-to-convert-gravimetric-soil-water-content-to-volumetric-soil-water-content/ (acedido em 23/08/2017).

[21] *Greenhouse Monitoring | Monnit Corp*. URL: http://www.monnit.com/ (acedido em 24/02/2017).

[22] W Guo, H Cheng, R Li, J Lü e H Zhang. "Greenhouse monitoring system based on wireless sensor networks". Em: *Nongye Jixie Xuebao/Transactions of the Chinese Society of Agricultural Machinery* 41.7 (2010), pp. 181–185. DOI: 10.1109/IITAW.2009.66.

[23] *HighCharts*. URL: https://www.highcharts.com/.

[24] *How to calibrate soil moisture sensors | Edaphic Scientific*. URL: http://www.edaphic.com.au/soil-water-compendium/soil-moisture-sensor-calibration/ (acedido em 23/08/2017).

[25] R. H. Hussain, A. F. Marhoon e M. T. Rashid. "Wireless Monitor and Control System for Greenhouse". Em: *International Journal of Computer Science and Mobile Computing* 2 (2013), pp. 69–87.

[26] maxim integrated. "DS1302 Trickle-Charge Timekeeping Chip CA 94086 408-737-7600". Em: (2015). URL: http://datasheets.maximintegrated.com/en/ds/DS1302.pdf (acedido em 02/08/2017).

[27] iteastudio. "I2C RTC/EEPROM Brick". Em: (2013). URL: ftp://imall.iteadstudio.com/Modules/IM130710004_I2C_RTC_EEPROM_Brick/DS_IM130710004.pdf.

[28] *Libelium - Homepage*. URL: http://www.libelium.com/ (acedido em 24/02/2017).

[29] J. Melorose, R. Perroy e S. Careas. *Arduino Nano*. 2015. DOI: 10.1017/CBO9781107415324.004. arXiv: arXiv:1011.1669v3.

[30] "MG811 CO2 Sensor". Em: ().

[31] *MicroSD card module for Arduino (SKU:DFR0229) - DFRobot Electronic Product Wiki and Tutorial: Arduino and Robot Wiki-DFRobot.com*. URL: https://www.dfrobot.com/wiki/index.php/MicroSD_card_module_for_Arduino_(SKU:DFR0229) (acedido em 02/08/2017).

[32] "Miniature I2C Digital Barometer". Em: (2012), pp. 1–17.

[33] J Morato, S Cruz, F Pereira e E. S. D. Tecnologia. "Multi-monitorização de estufa agrícola". Em: (2001).

[34] "MQ-2 Semiconductor Sensor for Combustible Gas". Em: *Pololu* (), p. 2.

[35] "Multilayer firewall system". Em: (1997).

[36] E. Naone. "Arduino Uno". Em: *Arduino Uni* 114.2 (2011), pp. 78–79. ISSN: 1098-6596. DOI: 10.1017/CBO9781107415324.004. arXiv: arXiv:1011.1669v3.

[37] *NOOBS for Raspberry Pi*. URL: https://www.raspberrypi.org/downloads/noobs/ (acedido em 07/09/2017).

[38] *Pinoccio - Wireless microcontroller for web-enabled DIY projects*. URL: https://pinocc.io/tech-specs (acedido em 10/09/2017).

[39] *Raspberry Pi 2 Model B - Raspberry Pi*. URL: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/ (acedido em 07/09/2017).

[40] *Raspberry Pi Downloads - Software for the Raspberry Pi*. URL: https://www.raspberrypi.org/downloads/ (acedido em 05/09/2017).

[41] *Raspberry Pi Foundation - About Us*. URL: https://www.raspberrypi.org/about/ (acedido em 07/09/2017).

[42] *RPi Hardware - eLinux.org*. URL: http://elinux.org/RPi_Hardware#cite_note-i2s-5 (acedido em 26/07/2017).

[43] *SANS - Information Security Resources*. URL: https://www.sans.org/security-resources/malwarefaq/pptp-vpn (acedido em 07/09/2017).

[44] N. Semiconductor. "Preliminary Product Specification v1.0". 2008.

[45] N. Semiconductors. "I2C-bus specification and user manual". Em: (2014).

[46] *Sensaphone - Homepage*. URL: https://www.sensaphone.com/ (acedido em 24/02/2017).

[47] *Sensohive*. URL: https://sensohive.com (acedido em 25/02/2017).

[48] M. Shang, G. Tian, L. Qin, J. Zhao, H. Ruan e F. Wang. "Greenhouse wireless monitoring system based on the ZigBee". Em: *IFIP Advances in Information and Communication Technology* 392 AICT.PART 1 (2013), pp. 109–117. ISSN: 18684238. DOI: 10.1007/978-3-642-36124-1_14.

[49] *SSH (Secure Shell) - Raspberry Pi Documentation*. URL: https://www.raspberrypi.org/documentation/remote-access/ssh/ (acedido em 07/09/2017).

[50] *Templated*. URL: https://templated.co/.

[51] *The Apache HTTP Server Project*. URL: https://httpd.apache.org/ (acedido em 07/09/2017).

[52] *Thermocouple Amplifier MAX31855 ID: 269: Adafruit Industries*. URL: https://www.adafruit.com/product/269 (acedido em 15/09/2017).

[53] *Thermocouple TMP36 Ardafruit Overview*. URL: https://learn.adafruit.com/thermocouple/overview (acedido em 15/09/2017).

[54] *UART*. URL: https://www.freebsd.org/doc/en/articles/serial-uart/.

[55] *Virtual Private Networking*. URL: https://msdn.microsoft.com/en-us/library/bb742566.aspx (acedido em 07/09/2017).

[56] *Win32 Disk Imager download | SourceForge.net*. URL: https://sourceforge.net/projects/win32diskimager/ (acedido em 05/09/2017).

[57] *Wireless Sensor Networks | LORD Sensing Systems*. URL: http://www.microstrain.com/wireless (acedido em 25/02/2017).

ANEXO I

ANEXOS

I.1 Resumos sobre o Estado da Arte

Nesta secção irão ser apresentados oito resumos sobre projetos académicos e dois quadros resumos sobre os artigos, estes resumos deram suporte ao capítulo 2 "Estado da Arte"secção "Síntese do Estado da Arte"desta dissertação. Nestes resumos está identificada as tecnologias utilizadas em cada um dos projetos comerciais e as aplicações quando usadas.

I.1.1 Projetos Científicos

I.1.1.1 Multi-monitorização de Estufa Agrícola

Este projeto foi desenvolvido em parceria com a Escola Superior de Tecnologia, Escola Superior Agrária e o Instituto Politécnico de Castelo Branco, e o seus autores são respetivamente J. Morato, S. Cruz, F. Pereira, J. C. Metrôlho e descreve um sistema desenvolvido pelos mesmos em que monitorizam as variáveis de temperatura e humidade de uma estufa, consulte o artigo [33]. O sistema desenvolvido é composto pelos sensores de temperatura e humidade que estão conectados a uma placa de aquisição de dados e por sua vez ligados a um computador com a capacidade de armazenamento numa base de dados. Posteriormente, o técnico poderá consultar a base de dados e verificar os parâmetros da estufa. Existem três hipóteses de visualização destes dados:

- Mensagens para o telefone móvel através da tecnologia "*Short Message Service*"(SMS);
- Acesso remoto à base de dados através da tecnologia "*Wireless Application Protocol*"(WAP);

- Visualização dos dados em tempo real num sitio web, utilizando a linguagem de programação ASP.NET pelo o lado do servidor.

I.1.1.2 *Greenhouse Monitoring with Wireless Sensor Network*

Este projeto foi realizado por Teemu Ahonen, Reino Virrankoski e Mohammed Elmusrati na "University of Vaasa"no "Department of Computer Science"em "Vaasa"na Finlândia, em que foi desenvolvido um sistema de monitorização de quatro variáveis de uma estufa, para mais tarde poderem controlar aquecedores, humidificadores, rega automática, etc. As variáveis estudas foram a temperatura, humidade, luz e dióxido de carbono, que são as mais importantes para crescimento das plantas. A arquitetura proposta implica que cada sensor tenha um rádio "*wireless*"e que comunique com um "*gateway*", que por sua vez está ligado a um PC para armazenar os dados dos sensores. Esta arquitetura permite que os sensores sejam deslocados em qualquer momento para outro local pretendido. A arquitetura foi testada e a conclusão retirada foi que os sensores têm um alcance de 10 [m] utilizando uma topologia tipo estrela com 5% de perda de pacotes. Isto dá-se devido à elevada concentração de humidade e da quantidade de vegetação existente na estufa. O sistema foi capaz de detetar os diferentes climas dentro da estufa [2].

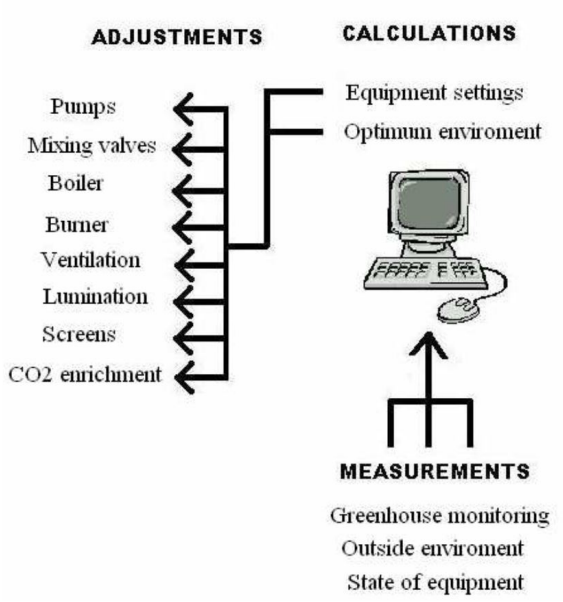

Figura I.1: Processos do Controlador dentro da estufa, adaptado de [2] da página 2

I.1.1.3 *Grennhouse Monitoring and Control System Based on ZigBee*

Mais recentemente, a China tem vindo a evoluir de uma agricultura tradicionalista para uma agricultura baseada em métodos mais modernos. Esta evolução pretendeu que este sector de atividade passasse a ser mais produtivo e de maior qualidade. Esta evolução foi

acompanhada do recurso à utilização de estufas e à introdução de sistemas de monitorização das variáveis que intervêm diretamente no processo produtivo, isto é, no crescimento das suas plantas, quer qualitativa quer quantitativamente. A monitorização aplicou-se à temperatura, humidade atmosférica, os níveis de concentração de dióxido de carbono presentes no ar e ainda a humidade no solo. A tecnologia usada foi a "ZigBee", que tornou possível o uso de comunicações por "Wi-Fi"e "Bluetooth". As vantagens da utilização desta tecnologia são: baixo nível de complexidade e baixo custo energético. Este projeto foi realizado por: Zhou Jianjun, Wang Xiaofang, Wang Xiu, Zou Wei e por Cai Jichen no "Beijing Research Center of Inteligent Equipment for Agriculture"e no "Beijing Academy of Agriculture and Forestry Sciences"[**Zhou2013**].

I.1.1.4 *Greenhouse Monitoring System Based on a Wireless Sensor Network*

A agricultura na Europa apresenta atualmente vários desafios. Há que tomar em consideração a escassez de água o elevado custo da energia e a baixa produtividade de produtos agrícolas o que tornam o sector pouco eficiente. A utilização de culturas em estufas, independentemente da sua localização, requerem sistemas de monitorização de variáveis como a temperatura do ar, a radiação solar e a humidade atmosférica, que são variáveis importantes para a qualidade do crescimento de produtos agrícolas. Em 2010 estimava-se que cerca 40% dos produtos agrícolas eram perdidos devido à falta de controlo. Como consequência do último foram desenvolvidas vários sistemas de monitorização de estufas com base na topologia WSN, ou seja, "*Wireless Sensor Network*". As redes de sensores, WSN, são redes quando bem estruturadas têm a capacidade de monitorizar e controlar os sistemas presentes nas estufas, minimizando assim o impacto da fraca produção no sector agrícola. A tecnologia usada é a "*Zigbee*"numa estrutura de rede de sensores, sendo este um dispositivo de comunicação, ou por "*Bluetooth*"ou por "*Wi-Fi*". Ao implementar o projeto, concluíu-se que existiam vários micro-climas dentro de uma estufa havendo grandes variações de temperatura ao longo da mesma, pelo que seria necessário ampliar o estudo a outras unidades de produção, para que os dados fossem mais representativos. Este projeto foi desenvolvido por: Ilias Lamprinos, Marios Charalambides e Michael Chouchoulis no "*Telco Software Department, INTRACOM TELECOM*"[22].

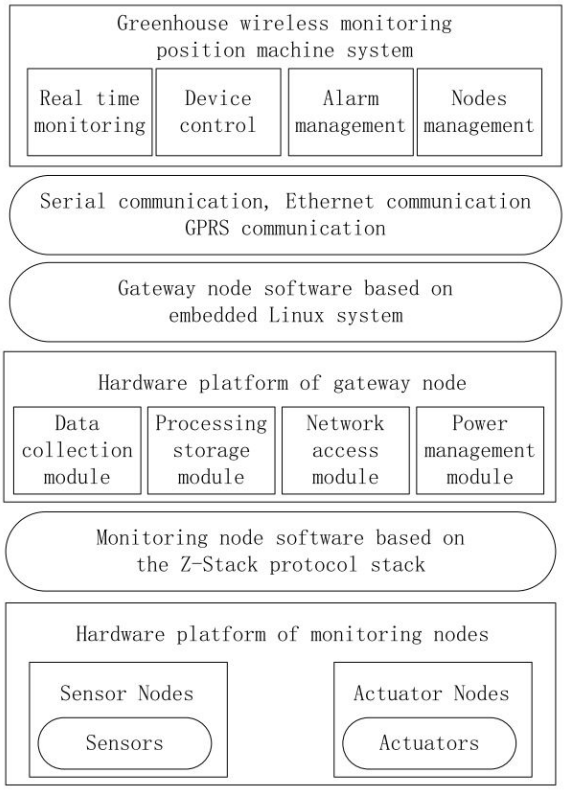

Figura I.2: Framework do sistema de monitorização de estufas, adaptado de [48] da página 3

O sistema é composto por três camadas. A primeira consiste nos vários pontos de comunicação, que têm a capacidade de sentir as variáveis a monitorizar e atuadores. Nesta fase todos os pontos de comunicação têm um dispositivo "ZigBee". Na segunda camada é enviado para um "gateway"toda a informação que os sensores medem. Finalmente, um computador dá o suporte necessário para armazenar dados e possibilita sua consulta. Este sistema é capaz de monitorizar as variáveis de interesse, guardar o dados dos sensores, controlar dispositivos de interesse, envio de informação para alertas e por fim verifica as comunicações existentes na rede, consulte o artigo [48] para informações mais detalhadas.

I.1.1.5 *Design of Greenhouse Enviroment Remote Monitoring System Based on Android Platform*

Um dos aspetos mais importantes da agricultura moderna é o uso de estufas. Nas estufas modernas é possível interligar a tecnologia de sensores com a tecnologia de comunicações sem fios, no sentido de criar um sistema de monitorização para o controlo das mesmas. Este sistema é compostos por várias fases. A primeira, todos os sensores têm um dispositivo "*ZigBee*"que comunica com um computador. O computador local gere a rede "*Bluetooth*"e armazena os dados. Seguidamente, a partir de um dispositivo 3G permite o envio da informação necessária para um "*Smartphone*". A aplicação móvel desenvolvida

para esse efeito opera num sistema operativo "*Android*".

Este projeto foi desenvolvido por Li Zhang, Congcong Li, Yushen Jia, Zhigang Xiao do "College of Mechanical & Electrical Engineering"e da "Agricultural University of Hebei"na China [18].

I.1.1.6 Wireless Monitor and Control System for Greenhouse

As estufas protegem as plantações das condições atmosféricas, o que permite o aumento da produção e a criação de condições ambientais propícias ao crescimento das plantas. Normalmente estas unidade de produção são mais utilizadas para a plantação de flores, de vegetais, de frutas e de tabaco. Os fatores básicos que afetam o desenvolvimento das plantas são a exposição à luz, a água, a temperatura, níveis de dióxido de carbono, etc. Estas variáveis são difíceis de controlar numa forma manual, o que torna os sistemas de monitorização e de controlo indespensáveis. As WSN, são cada vez mais relevantes, pois são capazes de comunicar entre si e têm várias aplicações, nomeadamente militares e comerciais. A tecnologia usada é a do "*ZigBee*". A arquitetura implementada consiste em que várias estufas que tenham um dispositivo "*ZigBee*"com os vários sensores ligados. Numa outra fase o dispositivo "*ZigBee*"comunica com um computador, que regista todos os dados recolhidos das unidades de produção [25].

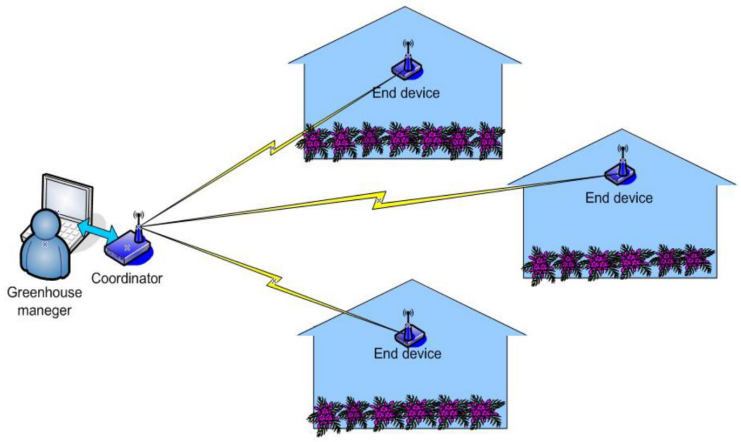

Figura I.3: Arquitetura do Sistema de Monitorização de várias estufas, adaptado de [25] da página 3

I.1.1.7 *Wireless Monitoring of the Green House System Using Embedded Controllers*

O desenvolvimento das plantas é afetado por vários fatores, que influenciam a qualidade da planta e o seu crescimento. São eles, a temperatura, a humidade e a exposição solar. A monitorização sistemática destas variáveis de ambiente permite recolher informação para melhor analisar o crescimento das plantas. Numa estufa ideal é possível o ajuste das variáveis atrás citadas de modo a incrementar a qualidade e a produção, além de diminuir os custos dos recursos utilizados tais como a água e a energia. O artigo aborda como

principal metodologia as WSN. O micro-controlador usado é o "*AT-Mega*". A vantagem da utilização de redes de sensores, WSN, é o facto de permitir que vários sensores sejam colocados nos locais apropriados e que comuniquem com um dispositivo central que armazena os dados. A arquitetura implementada tem as seguintes vantagens:

- Desenho da arquitetura robusta;
- Baterias que permitem a autonomia do sistema;
- Interface amiga do utilizador.

A arquitetura é composta pelos sensores, que sentem o ambiente que rodeia, que por sua vez estão conectados a um micro-controlador. Cada micro-controlador tem um dispositivo de comunicação Rádio-Frequência. A comunicação é feita para um dispositivo local, que analisa os dados e atua nos diversos dispositivos de controlo das variáveis de ambiente.

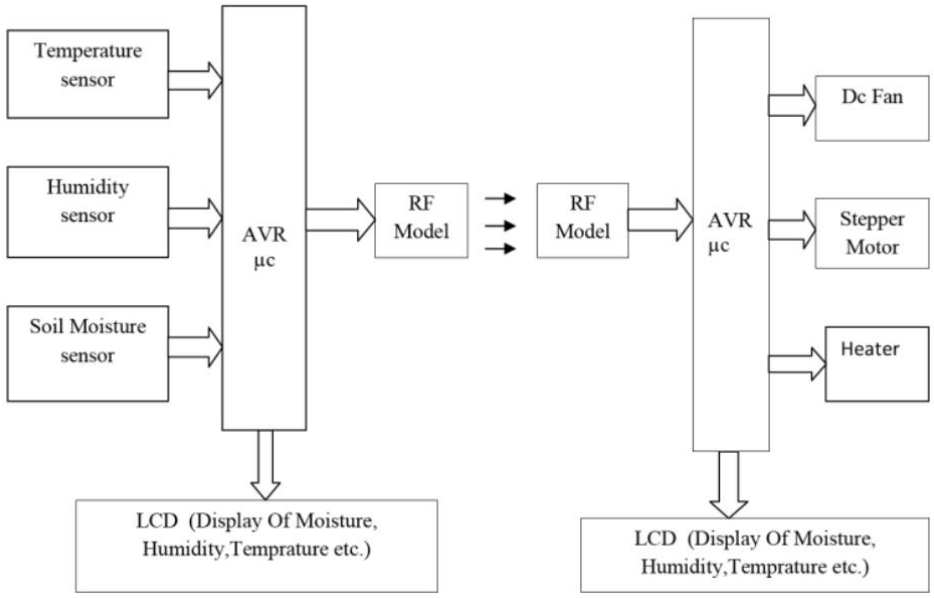

Figura I.4: Arquitetura do Sistema de Monitorização com Dispositivos RF, adaptado de [16] da página 2

Este projeto foi desenvolvido por Miss.Vrushali R. Deore e por Prof. V.M. Umale [16].

I.1.1.8 A Wireless Sensor Network Solution for Precision Agriculture Based on ZigBee Technology

Existem muitas aplicações para as redes de sensores. A grande vantagem que advém na utilização desta tecnologia é que a informação sentida pelos sensores pode ser enviada para outros pontos de comunicação. Essa informação pode ser analisada, guardada ou utilizada para controlar um sistema. As redes de sensores tem vindo a ser aplicadas em

monitorização de habitats, na agricultura, no controlo de reatores nucleares, na segurança e vigilância. O sistema desenvolvido neste projeto, descreve uma forma de desenvolver aplicações na agricultura, onde existe informação em tempo real de dados atmosféricos e outras variáveis ambientais. A arquitetura que foi desenvolvida consiste num conjunto de sensores que sentem o ambiente que os envolve e comunicam o seu estado com uma estação base. A norma que regula as redes de sensores é o 802.15.4. Este projeto foi desenvolvido por: Manijeh Keshtgari e por Amene Deljoo do "Department of Computer Engineering & IT, Shiraz University, Shiraz, Iran"e do "Department of E-Learning, Shiraz University, Shiraz, Iran", respetivamente [**Keshtgari2012**].

I.1.2 Quadros resumo dos artigos

Tabela I.1: Quadro de Resumo 1 de Tecnologias dos Artigos

Artigo	# Artigo	Monitoriza	Controla
Multi-monitorização de uma estufa agrícola	1	Sim	Sim
"Greenhouse Monitoring with Wireless Sensor Network"	2	Sim	Não
"Greenhouse Monitoring and Control System Based on ZigBee"	3	Sim	Sim
"Greenhouse Monitoring System Based on a Wireless Sensor Network"	4	Sim	Não
"Design of Geenhouse Enviroment Remote Monitoring System Based on Android Platform"	5	Sim	Sim
"Wireless Monitor and Control for Greenhouse"	6	Sim	Sim
"Wireless Monitoring of the Green House System Using Embedded Controllers"	7	Sim	Sim
"A Wireless Sensor Network Solution for Precision Agriculture Based on ZigBee Technology"	8	Sim	Sim

I.2 Arduino IDE

O "*Arduino IDE*"é uma aplicação multi-plataforma desenvolvida na linguagem de programação Java. Incluí um editor de código para a linguagem de programação em C/C++ capacitado para compilar e carregar os programas desenvolvidos para um micro-controlador "*Arduino*".

Como podemos observar na Figura: I.5 o botão com a marcação 1 tem a função de compilar o programa, o botão com a marcação 2 tem a função de compilar o programa

Tabela I.2: Quadro de Resumo 2 de Tecnologias dos Artigos

# Artigo	Comunicações	Aplicação
1	SMS e WAP	Site desenvolvido em ASP.NET
2	Rádio Wireless	-
3	ZigBee e GPRS	"Greenhouse wireless Monitoring Position Machine System"
4	ZiBee (Wireless e Bluettoth)	-
5	ZigBee e 3G	Aplicação Android parta "*Smartphone*"
6	ZigBee (WSN)	-
7	Rádio-Frequência (WSN)	-
8	Mesh	-

e carregar para o micro-controlador. A área de marcação 3 são os menus de opções da aplicação e por fim a área de marcação 4 é a área de edição do programa.

Nos menus de opções existem algumas funcionalidades importantes que permitem trabalhar melhor com a aplicação "*Arduino IDE*". Os seguintes pontos demonstram as funcionalidades e como as ativar. Podem ser consultados os sítios [**Arduino_IDE**, **Arduino_IDE_1**]:

- Novo Ficheiro: "*File -> New*";
- Abrir Ficheiro Existente: "*File -> Open*";
- Abrir exemplos providenciados pelas bibliotecas: "*File -> Examples -> Biblioteca -> Exmplo*";
- Guardar Ficheiro: "*File -> Save*";
- Guardar Ficheiros Como: "*File-> Save as*";
- Menu de Preferências: "*File -> Preferences*".;
- Sair da aplicação: "*File -> Quit*";
- Comentar código selecionado: "*Edit -> Comment*";
- Editar a indentação do código: "*Edit -> Increase Ident ou Edit -> Decrease Ident*;
- Descobrir palavra ou frase no código: "*Edit -> Find*";
- Descobrir próximo: "*Edit -> Find Next*";
- Descobrir anterior: "*Edit -> Find provious*";
- Verificar e compilar:"*Sketch -> Verify/Compile*";
- Carregar o programa para o micro-controlador: "*Skecth -> Upload*";
- Abrir a pasta do projeto: "*Sketch -> Show Sketch Folder*";

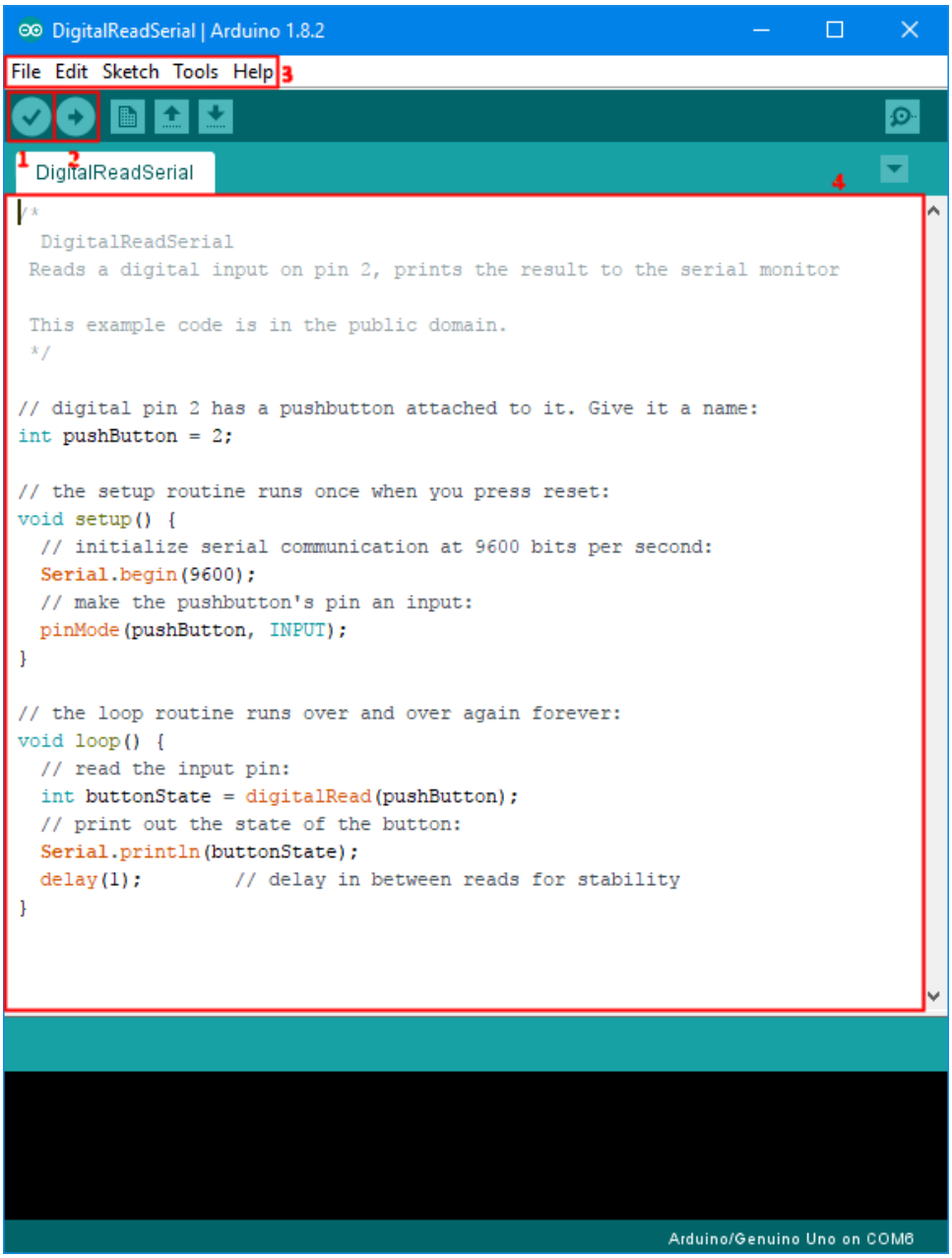

Figura I.5: Arduino IDE

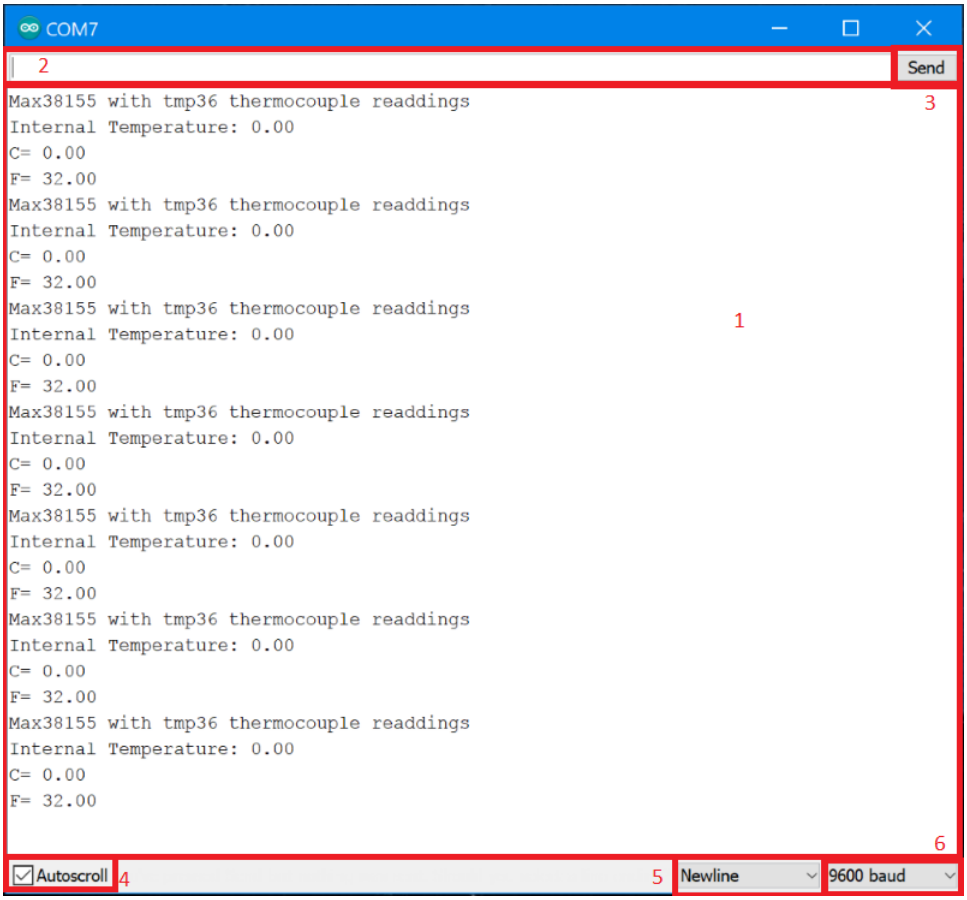

Figura I.6: Arduino IDE

- Incluir biblioteca no projeto: "*Sketch -> Include Library*";
- Incluir Ficheiro no projeto: "*Sketch -> Add File*";
- Aplicar indentação automática: "*Tools -> Auto Format*";
- Monitor de comunicação série: "*Tools -> Serial Monitor*";
- Monitor de desenho de gráficos: "*Tools -> Serial Plotter*";
- Actualização do "firmware": "*Tools -> WiFi101 Firmware*";
- Selecionar o micro-controlador: "*Tools -> Board*";
- Selecionar a porta usb: "*Tools -> Port*";
- A informação do micro-controlador: "*Tools -> Board Info*";
- Tipo de compilador: "*Tools -> Programmer*";
- Carregar o bootloadr: "*Tools -> Burn Bootloader*";
- Ajuda para começar a trabalhar com o arduino ide: "*Help*".

Antes de iniciar um projeto com a aplicação "*Arduino IDE*", é necessário identificar o tipo de micro-controlador "*Arduino*"no menu de opções "*Tools -> Board Info*". Seguidamente há que seleciona-lo no menu de opções "*Tools -> Board*"e selecionar a porta de comunicação no menu de opções "*Tools -> Port*".

I.2.0.1 Bibliotecas

As bibliotecas facultam a interligação entre o hardware e o software. A aplicação "*Arduino IDE*"viabiliza a instalação de novas bibliotecas, embora já existam algumas disponíveis. Esta secção irá explicar como instalar novas bibliotecas e como aceder às mesmas, já instaladas.

Para a instalação de novas bibliotecas na aplicação, deve-se proceder das formas a seguir indicadas. - No menu de opções "*Sketch -> Library -> Manage Library*", abre um novo menu onde existe uma lista de bibliotecas compatíveis com o micro-controlador "*Arduino*". Para instalar a biblioteca basta seleciona-la e pressionar no botão "*install*".

- A instalação de bibliotecas não presentes na lista tem um processo mais demoroso. Em primeiro lugar é necessário obter a biblioteca que pretende instalar, de seguida, em máquinas Windows e copiar os novos ficheiros na diretoria "*C:\Documentos\Arduino\librairies*".

No entanto, a aplicação "*Arduino IDE*"já tem algumas bibliotecas disponíveis para uso, que se situam na diretoria "*C:\Progrmas\Arduino\libraries*".

Normalmente as bibliotecas têm disponível para uso e consulta um programa pronto a ser executado, que explica de algum modo como elas funcionam e como aceder às funções que viabilizam uma integração otimizada do novo hardware ao micro-controlador "*Arduino*". Para aceder aos programas prontos a serem executados, basta selecionar o menu de opções "*File -> Examples -> [library] -> [file.ino or file.pde]*"[**Arduino_IDE_1**, **Biblioteca_2**].

Para mais informação pode ser consultada a secção "Programação: Área de Edição de Código"no §3 deste capítulo.

I.2.0.2 Serial Monitor

A Figura: I.6 é a janela de monitorização para a comunicação série 3.2.3.3. É uma interface de comunicação entre o micro-controlador "Arduino"e o utilizador, possibilita a visualização das comunicações estabelecidas e a construção de mensagens para transmitir ao micro-controlador "*Arduino*".

A Figura I.6, é a janela de monitorização para a comunicação série.

1. Painel de informação com o conteúdo das comunicações estabelecidas;
2. Caixa de texto que permite a construção de mensagens para transmitir ao micro-controlador "Arduino";
3. Botão "Send", faculta o envio do conteúdo da caixa de texto 2. Após a receção do conteúdo por parte do micro-controlador a informação exposta na caixa de texto é apagada;

4. Caixa de seleção "*Autoscroll*", permite que o painel de informação 1 acompanhe as novas comunicações, ocultando as comunicações mais antigas e simultaneamente expondo a nova informação;

5. O menu 4 apresenta 4 opções: - "*New Line*", que permite que no painel de informação 1, a nova linha seja impressa e no fim coloca o cursor no início da linha seguinte, em sistemas Windows. - "*Carriage Return*"possibilita que a nova linha seja impressa no painel de informação 1 e coloca o cursor no fim da mesma linha. - "*No Line Ending*"imprime a linha no painel de informação 1 e o cursor fica após o último caracter impresso. - "*Both NL & CR*"tem a mesma função que a opção "*New Line*", para qualquer sistema operativo. No entanto a opção "*New Line*", em alguns sistemas operativos, imprime a linha e o cursor passa para a linha seguinte, mas deixando para atrás espaços em branco na linha. O número de espaços livres na linha corresponde ao número de caracteres impressos na linha anterior;

6. Menu de velocidades de transmissão, viabiliza a escolha entre várias velocidades de transmissão entre o micro-controlador "Arduino"e o computador. As velocidades são 9600, 19200, 38400, 57600, 74880, 115200, 230400 e 250000 [baud rate].

I.2.1 Programação

A linguagem de programação a usar no "*Arduino IDE*"é o C/C++. Nesta secção irão ser apresentadas as rotinas e as funções que a aplicação "*Arduino IDE*"contém e como programar um micro-controlador "*Arduino*". Poderá visitar os sítios [**Arduino_Prog_1**, 7].

I.2.1.1 Área de Edição de Código

A programação no "*Arduino IDE*"é composta por 4 partes: - A adição de bibliotecas que possibilitam o funcionamento do micro-controlador "*Arduino*"com outros dispositivos periféricos. - A revisão da criação de variáveis globais e de construtores de acesso às funções existentes nas bibliotecas. - A inicialização de serviços e de terminais, caso estes sejam de "*output*"ou de "*input*". - A forma como é executado o programa e os cuidados a ter na sua programação.

Nesta secção irão ser explicados os vários conceitos, que facultam o suave funcionamento de um programa em execução num micro-controlador "*Arduino*".

A primeira parte, que é a inclusão de bibliotecas no projeto é bastante importante, pois permitem a integração de dispositivos periféricos à placa de aquisição de dados "*Arduino*". Como já foi referenciado na secção I.2, em primeiro lugar é necessário instalar a biblioteca, após a sua instalação é necessário incluir a biblioteca no projeto, no menu de opções "*Tools -> Library -> Add Library*". Ao executar este passo a biblioteca irá surgir na área de edição do código, a seguinte linha de código:

Listagem I.1: Inclusão de uma biblioteca no projeto

```
1  include <biblioteca.h>
2  ou
3  include "path/biblioteca.h"
```

Na terceira linha da Listagem: I.1 o uso de ("path/biblioteca.h") permite a utilização de bibliotecas fora das diretorias conhecidas pela aplicação "*Arduino IDE*". Enquanto na primeira linha (<biblioteca.h>), a aplicação "*Arduino IDE*"incluí as bibliotecas no projeto fazendo a pesquisa das mesmas em diretorias conhecidas pela aplicação.

A segunda parte é a criação de variáveis globais ou de construtores de acesso às funções existentes nas bibliotecas. A utilização de variáveis globais permite que o código seja simplificado e o seu uso, por vezes, permite que a sua execução seja mais rápida. A criação deste tipo de variáveis faz-se da seguinte forma:

Listagem I.2: Inclusão de uma biblioteca no projeto

```
1  #define intGlobal 1
2  #define floatGlobal 1.24
3  #define strGlobal Hello
4  #define boolGlobal True
```

Na Listagem I.2 não é aconselhável a frequente alteração do valor deste tipo de variáveis. A criação de variáveis não globais também permite incrementar a rapidez de execução do programa. A Listagem I.3 indica o modo de criação.

Listagem I.3: Variáveis não globais

```
1  int var=1;
2  float varfloat=1.24;
3  string strVar="Hello";
4  char charVar='H';
5  bool boolVar= false;
6  int ledPin=13;
7  int analogPin=A0;
```

Na linha seis e sete da Listagem I.3 possibilita a definição do terminal 13 do microcontrolador "*Arduino*"como uma variável designada "*ledPin*", assim como para o terminal analógico "A0"é uma variável "*analogPin*".

A terceira parte consiste em chamar as funções que inicialização os serviços, como podemos observar na Listagem: I.5.

Listagem I.4: setup

```
1  void setup(){
2    Serial.begin(9600);
3  }
```

Na Listagem: I.5, a função "Serial.begin(9600)"está inserida numa outra função designada "steup()". A importância desta função é compilar devidamente o programa. Já a função "*Serial.begin(9600)*"será abordada neste capítulo na secção 3.2.3.3.

A função "*loop()*", é uma função de ciclo infinito, ou seja quando o programa é executado e chega ao seu fim, o programa volta ao seu inicio e executa de novo o mesmo código. No entanto é necessário ter cuidado à forma como a execução desta função é efetuada. Quando esta função é extremamente densa, o uso do comando "*if*"ajuda a atenuar e até mesmo a evitar erros de execução.

Os erros execução podem ser da lentidão na execução do programa, ou até mesmo o programa estar continuamente a executar o mesmo código sem controlo, o que leva a um "*reset*"do micro-controlador "*Arduino*".

Por norma o código de um programa para um micro-controlador "Arduino"tem a seguinte forma:

Listagem I.5: setup

```
1  /*
2    DigitalReadSerial
3  Reads a digital input on pin 2, prints the result to the serial monitor
4
5  This example code is in the public domain.
6  */
7
8  // digital pin 2 has a pushbutton attached to it. Give it a name:
9  int pushButton = 2;
10
11 // the setup routine runs once when you press reset:
12 void setup() {
13     // initialize serial communication at 9600 bits per second:
14     Serial.begin(9600);
15     // make the pushbutton's pin an input:
16     pinMode(pushButton, INPUT);
17 }
18
19 // the loop routine runs over and over again forever:
20 void loop() {
21     // read the input pin:
22     int buttonState = digitalRead(pushButton);
23     // print out the state of the button:
24     Serial.println(buttonState);
25     delay(1);        // delay in between reads for stability
26 }
```

I.2.1.2 Comunicação Série

Quando se pretende desenvolver um programa, por vezes é necessário imprimir para o ecrã o resultado de algumas variáveis num certo momento da execução do mesmo, para o testar. No "*Arduino IDE*"para o fazer, usamos as funções do "*Serial*", que permite a comunicação através do USB para outro dispositivo. No entanto existem dois terminais no micro-controlador "*Arduino*", o Tx e o Rx que é outra forma de ligação para outros

dispositivos, caso estes não tenham interface USB. A comunicação série tem as seguintes funções:

- Serial.begin(speed): Função de inicialização da comunicação série. Tem como parâmetro de entrada a velocidade de transmissão [baud rate];
- Serial.end(), termina a comunicação em série;
- if(Serial), verifica se a comunicação já foi estabelecida;
- Serial.available(), verifica se existem na comunicação novos dados no "*shift register*";
- Serial.availableForWrite(), devolve o número de bytes disponíveis no "*shift register*";
- Serial.flush(), espera que o envio de dados termine;
- Serial.peek(), devolve o caracter seguinte do "*shift register*"sem o remover;
- Serial.print(), imprime no "*Serial Monitor*";
- Serial.println(),imprime no "*Serial Monitor*"e no fim passa para a linha seguinte;
- Serial.read(), lê o conteúdo do "*shift register*";
- Serial.readBytes(), retorna a dimensão da mensagem presente no "*shift register*";
- Serial.readBytesUntil(), devolve o número de carateres lidos dentro no "*shift register*";
- Serial.readString(), lê uma string do "*shift register*";
- Seral.readStringUntil(), lê os carateres do "*shift register*"e coloca-os numa string;
- Serial.setTimeout(), define o tempo de espera por uma nova comunicação;
- Serial.write(), escreve para o "*shift register*";
- serialEvent(), é uma função que é executada sempre se inicializa uma nova comunicação.

I.2.1.3 Definição dos terminais Digitais e Analógicos

Nesta secção irão ser apresentados métodos de como realizar leituras de dispositivos externos, de como enviar dados e utilizar as funções predefinidas.

Existem dois tipos de terminais num micro-controlador "*Arduino*", o terminal Digital e o terminal Analógico. Fisicamente ambos são sensíveis à tensão. No entanto, em software quando um terminal digital está em "*HIGH*", significa que está a gerar 5 [V] ou a receber 5 [V], consoante as funções de controlo do terminal estão a atuar ou a ler, respetivamente. Já o comando "*LOW*"indica que o terminal está a gerar 0 [V] ou a receber 0 [V].

As funções que permitem funcionar com os terminais digitais são:

- pinMode(13,OUTPUT), define que o terminal 13 do micro-controlador "Arduino"é um output;
- pinMode(13,INPUT), define que o terminal 13 do micro-controlador "Arduino"é um input;
- digitalRead(13), devolve o estado do terminal caso seja "HIGH"ou "LOW";
- digitalWrite(13, HIGH), define o terminal como "HIGH", isto é está a gerar uma tensão de 5 [V];
- digitalWrite(13, LOW), define o terminal como "LOW", isto é está a gerar uma tensão de 0 [V].

Listagem I.6: exemplo de uso do terminal digital

```
1  #define ledPin 13;
2
3  void setup() {
4    pinMode(ledPin, OUTPUT);
5    digitalWrite(ledPin, HIGH);
6  }
7
8  void loop() {
9    if(digitalRead(13,HIGH)){
10     digitalWrite(ledPin,LOW);
11   }else{
12     digitalWrite(ledPin,HIGH);
13   }
14 }
```

As portas analógicas permitem efetuar leituras e gerar tensões de valores entre 0 [V] e 5 [V]. As funções associadas a este tipo de terminal são:

- "*val=analogRead(A0)*", devolve um valor em bytes convertido para decimal. Para calcular a equivalente tensão basta $val * 5/1024$;
- "*analogWrite(A0,value)*", gera uma tensão no terminal A0.

I.3 Comunicações

As comunicações são extremamente importantes pois permitem a partilha de dados. Neste caso os dados são relativos a sensores e estados de operação entre micro-controladores. Nesta secção irão ser abordados os temas de comunicação "*Mesh*"e "*Bluetooth*"que irão ser as tecnologias de comunicação implementadas neste projeto.

I.3.1 Mesh

As redes sem fios "*Mesh*"foram originalmente desenvolvidas para aplicações militares, hoje em dia a sua aplicação serve para cobrir áreas de grande dimensão usando apenas uma ligação à Internet. Consequentemente, é necessário uma melhor distribuição da largura de banda. A solução encontrada foi a criação de vários pontos de acesso, designados "*Nodes*"ou por "*Hops*", espalhados pela área que se pretende cobrir. Foram desenvolvidos três arquiteturas para as redes sem fios "*Mesh*":

1. - A 1ª geração, "*Radio ad hoc*", utiliza apenas um canal para o serviço de clientes e para o serviço de "*Mesh Backhaul*". Esta arquitetura é a que apresenta a pior distribuição da largura de banda.

2. - A 2ª geração, "*Radio Meshed Backhaul*", utiliza dois rádios. Um dos rádios realiza o serviço de clientes, pode seguir o standard 802.11b/g e utiliza uma largura de banda a 2.4 [GHz]. O outro rádio cria uma rede "*Mesh*", segue o standard 802.11a e utiliza uma largura de banda a 5.8 [GHz]. Esta arquitetura apresenta uma melhor performance do que a primeira arquitetura, pois usa dois rádios para serviços distintos em larguras de banda diferentes. No entanto a ligação ao serviço de Internet faz-se no mesmo canal o que leva a uma degradação da comunicação.

3. - A 3º geração, "*Radio Structured Mesh*", utiliza três rádios. Um dos rádios serve para a criação de uma rede "*Mesh Backhaul*", os outros dois servem para o serviço de clientes, dos quais um serve para o serviço de "*downlink*"e outro para o serviço de "*uplink*". Esta arquitetura é a que apresenta a melhor performance pois utiliza três rádios para funções distintas a operarem em canais com frequências distintas que não interferem entre si. Consulte o sítio [**Mesh**].

I.3.2 Bluetooth

A tecnologia "*Bluetooth*"é composta por uma rede sem fios de baixo consumo energético que é utilizado na partilha de informação entre dispositivos. Esta tecnologia tem dois ramos de aplicações, a tecnologia "*Bluetooth BR/EDR*"e a tecnologia "*Bluetooth Low Energy (BT LE)*". O "*Bluetooth Basic Rate / Enhanced Data Rate (BR/EDR)* possibilita a conexão continua entre dispositivos utilizando a topologia "*point-to-point (p2p)*". Informação detalhada na secção "Point-to-Point (P2P)"I.3.2.1.

A tecnologia "*Low Energy*"realiza conexões "*short-burst*"em redes sem fios utilizando várias topologias de rede, tais como P2P, "*Broadcast*"e "*Mesh*".

A topologia P2P I.3.2.1 para o BT LE otimiza a transferência de dados. As aplicações desta topologia são, a conexão de dispositivos de monitorização de saúde e rastreadores de aptidão física.

A topologia BT LE "*Broadcast*"suporta a partiha de informação localizada. Esta topologia deu origem à tecnologia "*Beacons*"que consiste na partilha de informação localizada.

A tecnologia BT LE "*Mesh*"utiliza a topologia (m:m) I.3.2.2, foi otimizado para a criação de redes sem fios de grandes dimensões e é ideal para a automação de edíficios, redes de sensores e soluções de localização.

A tecnologia "*Bluetooth*"opera na largura de banda de 2.4 [GHz], ou seja, 2.4000 a 2.4835 [GHz]. Na tecnologia "*Bluetooth BR/EDR*"a velocidade de transmissão de dados está compreendida entre 1 [Mb/s] a 3 [Mb/s] dependendo da modulação utilizada. Informação exposta nos sítios [**BT_Core5**] e [**BT**] . O standard que a tecnologia "*Bluetooth*"se rege é o 802.15.1 desenvolvido pela IEEE.

I.3.2.1 Topologia "*Point-to-Point*"(P2P)

A topologia P2P viabiliza a conecção entre dois dispositivos "*Bluetooth*". Para mais informações visite o sítio [**BT**].

I.3.2.2 Topologia "*Many-to-Many*"(m:m)

A topologia m:m possibilita que vários dispositivos com a tecnologia BT LE "*Mesh*"se conectem à mesma rede. Informação exposta no sítio [**BT**].

I.4 Módulos e Sensores

Nesta secção estão apresentados módulos e sensores que foram estudados para a implementação da componente prática da dissertação, mas no entanto não foram usados por existirem alternativas que apresentam mais vantagens. No caso do sensor MQ2 apenas estão apresentados os gráficos que deram a origem às funções de transferência, a restante informação encontra-se no capítulo 4 "Implementação"na secção "MQ2"4.3.3.1. A secção "Modos de Operação do Rádio nRF24L01+"pretende apresentar os Modos de funcionamento do rádio nRF24L01+, no entanto este conteúdo não foi desenvolvido na componente prática desta dissertação.

I.4.1 Modos de Operação do Rádio nRF24L01+

A Tabela 3.26 apresenta os valores da corrente de alimentação dos modos existentes. Existem 5 modos de operação da placa de comunicação nRF24L01+, que são "*Power Down*", "*Standby I*", "*Standby II*", "*RX*"e o "*TX*", que afetam o desempenho do componente.

1. O modo "*Power Down*"é um processo de desativação da placa de comunicação, no entanto este modo não apaga valores nem registos que foram anteriormente gravados e o protocolo de comunicação SPI contínua ativo. A corrente de alimentação neste modo é de 900 [nA], que é a menor corrente que este componente consome. Para dar o início ao processo de desativação da placa de comunicação é necessário colocar o bit "PWR_UP"a zero no registo de configuração.

2. O modo "*Standby I*"foi desenhado para minimizar a corrente média consumida enquanto mantém a operação de comunicação. O controle da corrente consumida é feita devido a uma parte do oscilador estar desativado. Para ativar este modo de operação é necessário colocar o bit "PWR_UP"a 1 no registo de configuração.

3. O modo "*Standby II*"dá acesso aos "*buffers*"extra do clock e ativa todas as partes do oscilador.

4. O modo "*RX*"e o modo "*TX*", são considerados os modos ativos, pois são estes modos que permitem as comunicações dentro da mesma rede "*Mesh*". Estes podem operar em simultâneo com os modos "*Standby I*"e "*Standby II*". O modo "*RX*"ativa o processo que receção de pacotes emitidos por uma outra placa de comunicações. Para ativar este modo é necessário colocar os bits "PWR_UP", "PRIM_RX"e o terminal "CE"ativos a 1.

5. O modo "*TX*"ativa o processo de transmissão de pacotes. Para ativar este modo é necessário colocar os bits "PWR_UP"e o "PRIM_RX"a zero e o terminal "CE"necessita de estar ativo durante 10 [μs].

I.4.2 BT-HC05

A placa de comunicação BT HC-05 cria uma rede sem fios "*Bluetooth V2.0+EDR (Enhanced Data Rate)*"I.3.2. Este módulo, dependendo da modulação, permite uma velocidade de transmissão de dados a 3 [Mbps] e opera na banda 2.4 [GHz].

Este módulo tem um modo de baixa potência, com uma tensão de alimentação a 1.8 [V], no entanto permite valores de tensão compreendidos entre 1.8 a 3.6 [V]. A potência de transmissão é de +4 [dBm] e contêm uma antena integrada. Este permite ainda interligar com um micro-controlador externo usando o protocolo de comunicação "UART"3.2.3.4. O módulo BT HC-05 tem a capacidade de se conectar novamente ao último dispositivo "*Bluetooth*"conhecido, em caso de falha, possibilita ainda a definição a um dispositivo para emparelhamento por defeito. A password de acesso é "0000"ou "1234". Informação detalhada no datasheet [**BT_HC-05**]. A Tabela: I.3 indica as especificações técnicas, a Tabela: I.4 apresenta a configuração do protocolo de comunicação "UART"e por fim a Tabela:I.5 demonstra os terminais do módulo BT HC-05 e suas funções.

I.4.2.1 Interface com o micro-controlador Arduino

Visto que a placa de comunicação BT HC-05 cria uma ligação a um micro-controlador externo via protocolo de comunicação "UART"é necessário garantir que os terminais estejam ligados corretamente. A Tabela I.6 apresenta a forma.

A programação que gere as comunicações entre a placa de comunicação BT HC-05 e o micro-controlador "Arduino"segue o standard de comunicação série do Arduino, pondere sobre este assunto na secção "Programação: Comunicação Serie"I.2.1.2 deste capítulo.

Tabela I.3: Especificação Técnica do módulo BT-HC05

Descrição	BT HC-05
Protocolo de Bluetooth	BT 2.0+EDR
Frequência [GHz]	2.4
Modulação	GFSK
Potência de Emissão [dBm]	Classe 2
Potência de Transmissão[dBm]	<=-84 a 0.1% Erro
Velocidade Assincrona [Mbps]	2.1
Velocidade Sincrona [Kbps]	160
Tensão de entrada [V]	3.3
Corrente de entrada [mA]	50
Temperatura em funcionamento[°C]	-20 até 55
Master	Sim
Slave	Sim
Comandos AT	Sim

Tabela I.4: Configuração do protocolo de comunicação UART para o módulo BT-HC05

"*Data bits*"	8
"*Stop bit*"	1
Paridade	Não
Velocidades de Transmissão	9600, 19200, 38400, 57600, 115200, 230400, 460800

Tabela I.5: Terminais do módulo BT-HC05

Terminal	Nome	Função
1	"*Key*"	Terminal que permite selecionar o modo de comunicação ou o modo de comandos AT
2	V_{CC}	Terminal de massa comum
3	GND	Terminal de alimentação
4	TXD	Terminal de transmissão
5	RXD	Terminal de receção
6	"*State*"	Terminal de Output para identificação do estado

I.4.3 DHT22

O sensor DHT22, é um sensor digital que determina os valores de temperatura e de humidade do meio onde se insere. Este sensor é constituído por um único bus de comunicação e uma memória OPT ou PROM. O seu sinal digital de saída foi devidamente calibrado num laboratório certificado para esse efeito. A gama de valores de saída de fábrica do sensor foram guardados na PROM, para que seja possível criar a função de calibração. Este sensor tem baixo consumo energético, é compacto e de pequenas dimensões e ainda possibilita ligações por cabo até 20 metros. Informação exposta no datasheet [**Liu**].

Tabela I.6: Conexão entre os terminais do módulo BT HC-05 e o Arduino

Terminal BT HC-05	**Micro-Controlador Arduino**
TXD	RX
RXD	TX
V_{CC}	3.3V
GND	GND

As aplicações para o sensor DHT22 são medições de humidade relativa do ar e medições de temperatura. É usado para conexões p+or cabo de longas distâncias.

I.4.3.1 Especificações Técnicas

O sensor DHT22 tem uma tensão de alimentação em DC entre 3.3 e 6 [V]. Existem duas variantes deste sensor, uma de pequena dimensões, 14x18x5.5 [mm] e um outro com 22x28x5 [mm]. Este sensor também permite a ligação de um cabo até 20 [m] de comprimento, o que torna a sua utilização ótima para grandes distâncias. A Tabela: I.7 apresenta as características do sensor de humidade e temperatura DHT22.

Tabela I.7: Especificações do sensor DTH22

	Humidade Relativa	**Temperatura**
Gama de saída	0 a 100 [%RH]	-40 a 80 [0°C]
Precisão	± 2 %[RH]	$< \pm 0.5$ [°C]
Resolução	0.1 [%RH]	0.1 [°C]
Repetibilidade	$\pm 1\%[RH]$	$\pm 0.2[C]$
Histerese	$< \pm 0.3\%[RH]$	-
Estabilidade a longo prazo	$\pm 0.5[RH]/ano$	-

O sensor é composto por quatro terminais e a Tabela: I.8 indica a função de cada um deles.

Tabela I.8: Terminais DHT22

Terminal	**Descrição**
1	Tensão de Alimentação em de 3.3 a 6 [V]
2	Data Signal, um único bus de comunicação
3	NULL
4	Ground

I.4.4 MQ2

Neste gráfico podemos observar que não existe a possibilidade de calcular a concentração do ar, pois como tal a linha de tendência é uma linha vertical. A função de transferência

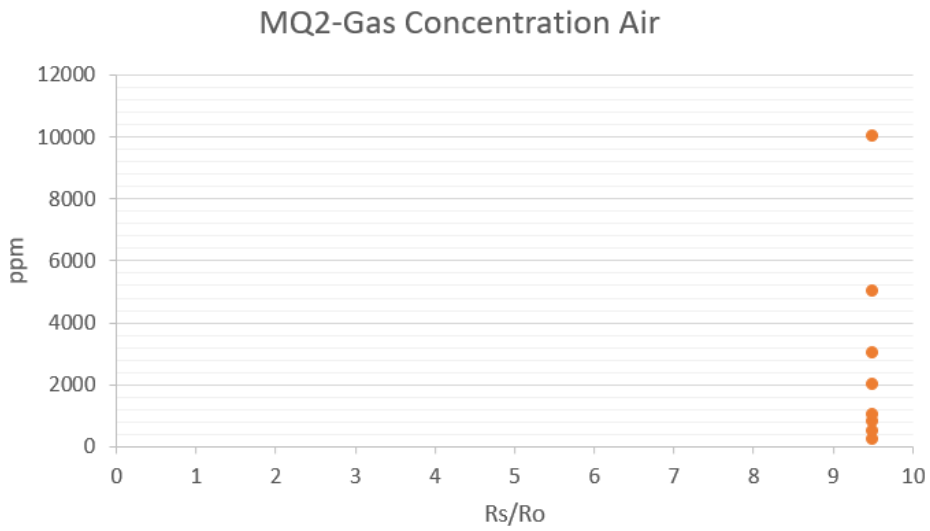

Figura I.7: Gráfico Rs/Ro por ppm do Ar para o sensor MQ2

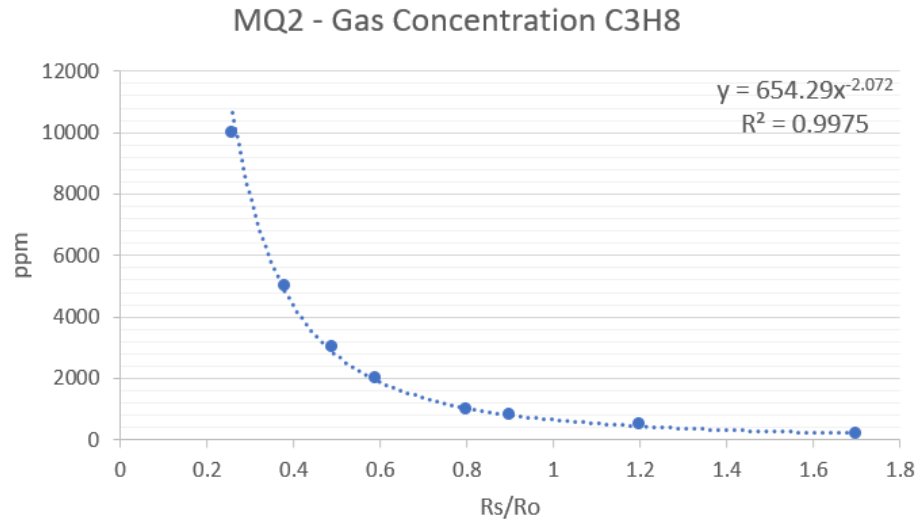

Figura I.8: Gráfico Rs/Ro por ppm do Propano para o sensor MQ2

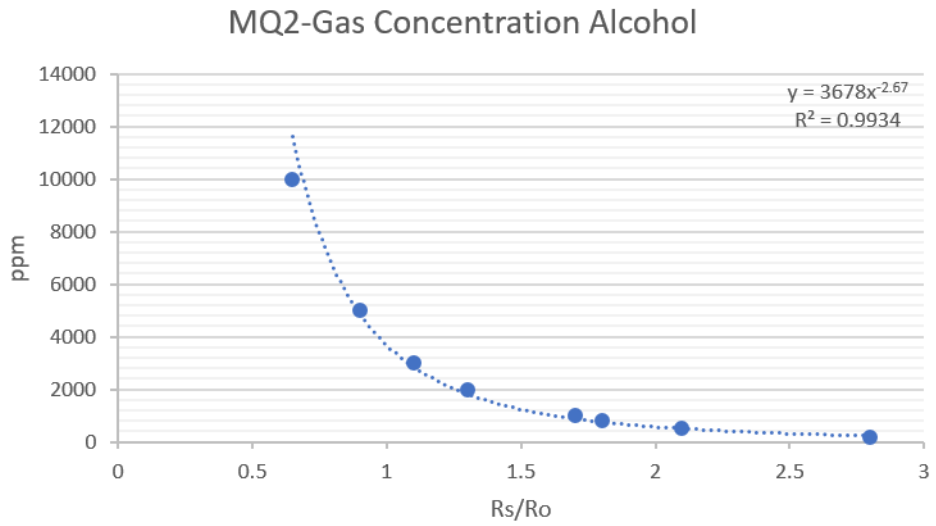

Figura I.9: Gráfico Rs/Ro por ppm do Álcool para o sensor MQ2

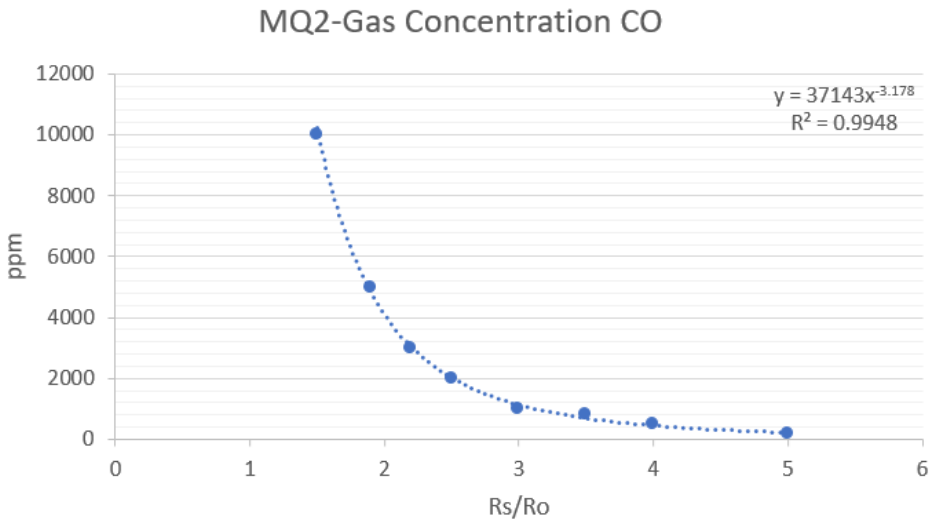

Figura I.10: Gráfico Rs/Ro por ppm do gás Monóxido de Carbono para o sensor MQ2

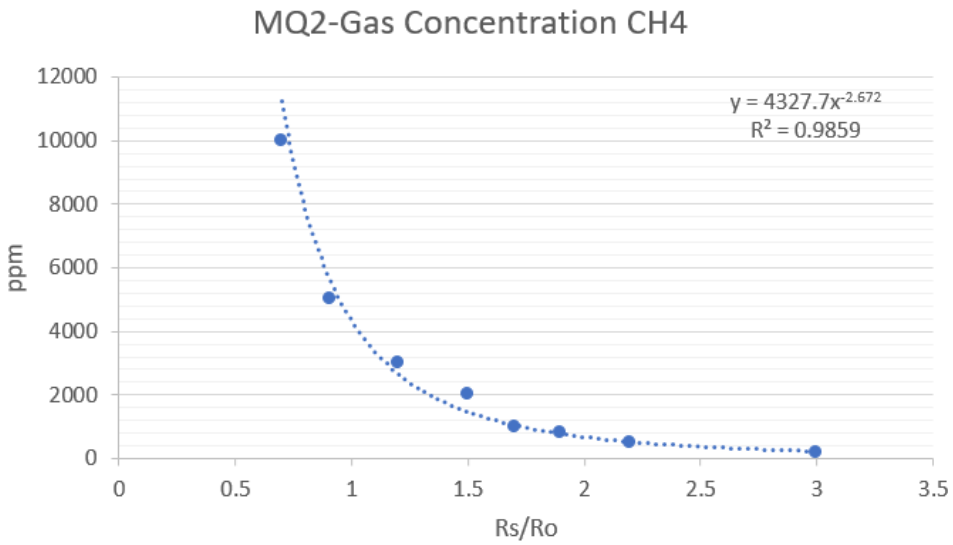

Figura I.11: Gráfico $\frac{R_s}{R_o}$ por [ppm] do gás Metano para o sensor MQ2

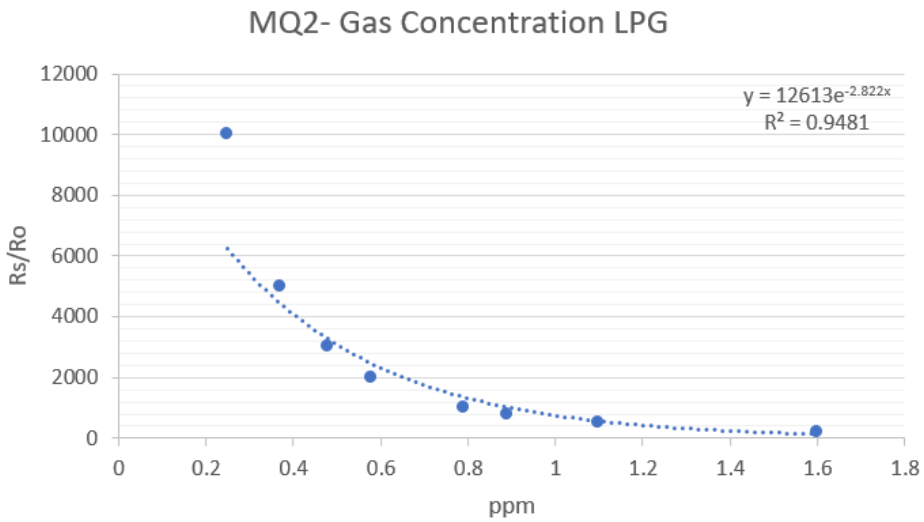

Figura I.12: Gráfico Rs/Ro por ppm do gás GPL para o sensor MQ2

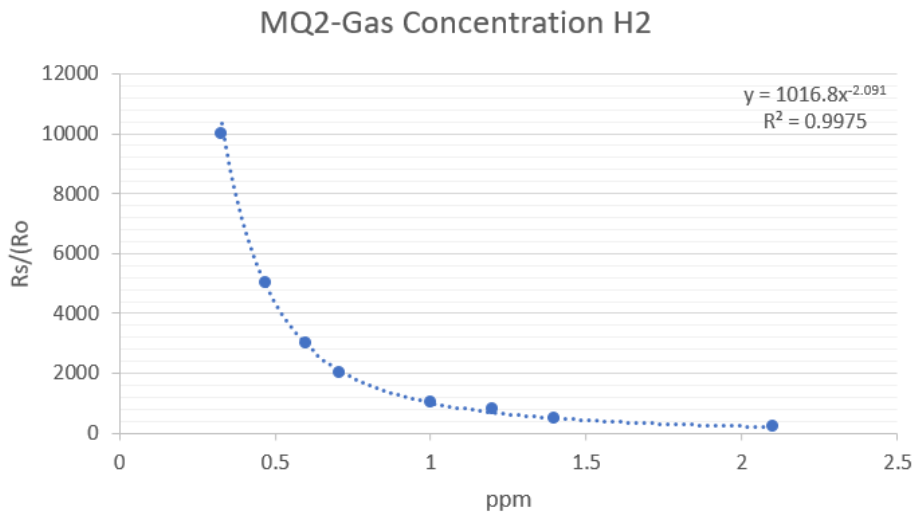

Figura I.13: Gráfico Rs/Ro por ppm do gás H2 para o sensor MQ2

gerada será:

$$\frac{R_S}{R_O} = 9.5 \tag{I.1}$$

Erro quadrático:

$$R^2 = 1 \tag{I.2}$$

Os pontos escolhidos para calcular a função de transferência do gás propano originou o gráfico expresso na Figura: I.8.

Neste gráfico é possível observar que a razão $\frac{R_S}{R_O}$ varia entre:

$$0 < \frac{R_S}{R_O} < 2 \tag{I.3}$$

A concentração [ppm] varia:

$$0 < C_{c3h8} < 10400[ppm] \tag{I.4}$$

A linha de tendência usada foi a de potência e a função de transferência obtida foi:

$$C_{c3h8} = 654.290 \frac{R_S}{R_O}^{-2.072} [ppm] \tag{I.5}$$

O erro quadrático da função de transferência I.5:

$$R^2 = 0.9975 \tag{I.6}$$

Os pontos escolhidos para calcular a função de transferência do gás Álcool originou o seguinte gráfico.

Neste gráfico podemos observar que que a razão entre $\frac{R_S}{R_O}$:

$$0 < \frac{R_S}{R_O} < 3 \tag{I.7}$$

A concentração do gás Álcool varia:

$$0 < C_{alcool} < 10800[ppm] \tag{I.8}$$

A linha de tendência usada foi a de potência e a função de transferência obtida é:

$$C_{alcool} = 3678 \frac{R_S}{R_O}^{-2.67} [ppm] \tag{I.9}$$

O erro quadrático obtido é

$$R^2 = 0.9934 \tag{I.10}$$

Os pontos escolhidos para calcular a função de transferência do gás monóxido de carbono originou o seguinte gráfico.

Neste gráfico podemos observar que $\frac{R_S}{R_O}$ varia:

$$1.5 < \frac{R_S}{R_O} < 5 \tag{I.11}$$

A concentração do gás Álcool varia:

$$0 < C_{co} < 10000[ppm] \tag{I.12}$$

A linha de tendência usada foi a potência e a função de transferência obtida é:

$$C_{co} = 37143 \frac{R_S}{R_O}^{-3.178} [ppm] \tag{I.13}$$

O erro quadrático obtido é

$$R^2 = 0.9948 \tag{I.14}$$

Os pontos escolhido para calcular o gás Metano originou o seguinte gráfico.

Neste gráfico podemos observar que a razão $\frac{R_S}{R_O}$ varia:

$$0.5 < \frac{R_S}{R_O} < 3 \tag{I.15}$$

A concentração do gás Metano varia:

$$0 < C_{metano} < 10000[ppm] \tag{I.16}$$

A linha de tendência usada foi a potência e a função de transferência obtida é:

$$C_{metano} = 4327.7 \frac{R_S}{R_O}^{-2.672} [ppm] \tag{I.17}$$

O erro quadrático obtido é

$$R^2 = 0.9859 \tag{I.18}$$

Os pontos escolhidos para do gás de petróleo liquefeito (GPL) originou o seguinte gráfico.

Neste gráfico podemos observar que a razão $\frac{R_S}{R_O}$ varia:

$$0.25 < \frac{R_S}{R_O} < 1.6 \tag{I.19}$$

A concentração do gás GPL varia:

$$0 < C_{gpl} < 4700[ppm] \tag{I.20}$$

A linha de tendência usada foi a exponencial e a função de transferência obtida é:

$$C_{gpl} = 12613e^{-2.882\frac{R_S}{R_O}}[ppm] \tag{I.21}$$

O erro quadrático obtido é

$$R^2 = 0.9481 \tag{I.22}$$

Apesar do erro quadrático estar próximo de 1, a linha de tendência não é ótima, mas para este caso a melhor, pois o ponto (0.5;10000) não esta incluído na função de transferência.

Os pontos escolhidos do gás Hidrogénio originou o seguinte gráfico.

Neste gráfico podemos observar que a razão $\frac{R_S}{R_O}$ varia:

$$0.3 < \frac{R_S}{R_O} < 2.2 \tag{I.23}$$

A concentração do gás Hidrogénio varia:

$$0 < C_h < 10000[ppm] \tag{I.24}$$

A linha de tendência usada foi a potência e a função de transferência obtida é:

$$C_h = 1016.8 \frac{R_S}{R_O}^{-2.091} [ppm] \tag{I.25}$$

O erro quadrático obtido é

$$R^2 = 0.9975 \tag{I.26}$$

I.4.5 DS1302

O circuito integrado DS1302 é um RTC e contém um circuito interno que faculta a conceção de uma bateria externa. Este RTC faz a gestão de todas as funções do tempo 3.5.2, opera a uma tensão de alimentação em DC entre 2.0 a 5.5 [V] e tem uma corrente de 300[nA] a 2.0 [V]. Contém 31 Byte de RAM o que proporciona o armazenamento de dados relativos ao tempo [26] . Este circuito integrado contém 8 terminais e as suas funções estão apresentadas na Tabela: I.9.

Tabela I.9: Terminais do módulo RTC DS1302

Terminal	Descrição
V_{CC2}	Terminal de alimentação do circuito integrado DS1302
X1 e X2	Terminais para conectar um "*Clock*"32.768 [KHz] de Quartzo
GND	"*GND*"
CE	"*Chip Enable*", permite que as funções de leitura ou de escrita sejam ativas.
I/O	Terminal bidirecional que permite que as funções de leitura e escrita sejam executadas.
SCLK	Terminal de entrada do "*clock*" do dispositivo externo.
V_{CC1}	Terminal de entrada que permite uma bateria externa seja conectada ao circuito integrado para alimenta-lo quando V_{CC2} está a 0 [V].

I.5 Comandos Linux

Os sistemas Linux estão cada vez mais desenvolvidos para ambientes gráficos e permitem um acesso fácil à máquina. Os comandos Linux operam num ambiente terminal. Nesta

secção irão ser apresentados os comandos Linux na distribuição "*Debian*"mais importantes, informe-se [**linuxcommand**].

- sudo - permite que o comando que se segue seja executado com permissões de administrador;
- sudo apt-get update - permite realizar uma atualização às tabelas de pacotes do sistema;
- sudo apt-get upgrade - permite realizar a atualização dos pacotes do sistema;
- sudo apt-get update && sudo apt-get upgrade - permite na mesma instrução executar dois comandos, separados por "&&";
- sudo apt-get update && sudo apt-get upgrade -y - O último comando permite atualizar automaticamente sem parar para que o utilizador responda afirmativamente à atualização;
- cd [diretoria] - permite mudar de diretoria;
- cd.. - Muda para diretoria "\home\usr\";
- ls - Comando que lista os conteúdos da diretoria;
- pwd - Comando que disponibiliza a informação da diretoria de trabalho;
- mkdir - Comando que cria uma nova diretoria;
- rm - Comando que permite apagar diretorias;
- cp - Comando para copiar diretorias e ficheiros;
- mv -Comando que permite mover diretorias e ficheiros;
- chmod - Comando que permite dar permissões a um ficheiro ou diretoria;
- cgown - Comando que permite mudar o dono ou grupo de um ficheiro ou diretoria;
- ifconfig - Comando que permite observar as configurações dos dispositivos de rede;
- nano - Comando para abrir o editor de texto nano.

I.6 Putty

O programa Putty é um cliente para ligações telnet ou SSH 3.7.4. Este programa serve para criar uma ligação remota de trabalho, consulte [**putty**].

I.7 Notepad++

O programa Notepad++ é um editor de texto útil para editar ficheiros de programas. O editor de texto notepad++ foi escrito na linguagem de programação C/C++ e é possível adicionar "plugins"que permitem tornar o editor de texto mais completo e facilmente escalável, informe-se [**npp**].

I.7.1 Plugin Npp FTP

Um plugin é um programa extra que opera em conjunto com o programa-mãe.

O plgin Npp FTP, conecta-se a uma máquina na mesma rede pelo protocolo File Transfer Protocol, ou seja realiza o "*download*"e o "*upload*"de ficheiros para outra máquina, informe-se no sítio [**nppFtp**].

I.8 Conteúdos de Suporte ao capítulo 4

Nesta secção irão ser apresentadas vários conteúdos que dão suporte ao capítulo 4.

Listagem I.7: Função em ciclo que permite a leitura da mensagem recebida

```
void loop(void){
  // Check the network regularly
  network.update();

  // Is there any
  thing ready for us?
  while ( network.available() ) {
     // If so, grab it and print it out
    RF24NetworkHeader header;
    payload_t payload;
    network.read(header,&payload,sizeof(payload));
    Serial.print("Received␣packet␣#");
    Serial.print(payload.counter);
    Serial.print("␣at␣");
    Serial.println(payload.ms);
  }
}
```

Listagem I.8: Função ciclo que permite a transmissão de uma mensagem

```
void loop() {
  // Check the network regularly
  network.update();
  // If it´s time to send a message, send it!
  unsigned long now = millis();
  if ( now - last_sent >= interval  ){
    last_sent = now;
    Serial.print("Sending...");
    payload_t payload = { millis(), packets_sent++ };
```

```
10      RF24NetworkHeader header(/*to node*/ other_node);
11      bool ok = network.write(header,&payload,sizeof(payload));
12      if (ok)
13        Serial.println("ok.");
14      else
15        Serial.println("failed.");
16    }
17  }
```

Listagem I.9: Registo de endereço de um Hop

```
1   void loop() {
2     // Check the network regularly
3     network.update();
4
5     // If it's time to send a message, send it!
6     unsigned long now = millis();
7     if ( now - last_sent >= interval  ){
8       last_sent = now;
9
10      Serial.print("Sending...");
11      payload_t payload = { millis(), packets_sent++ };
12      RF24NetworkHeader header(/*to node*/ other_node);
13      bool ok = network.write(header,&payload,sizeof(payload));
14      if (ok)
15        Serial.println("ok.");
16      else
17        Serial.println("failed.");
18    }
19  }
```

Listagem I.10: Função de sincronização da informação na rede

```
1     network.update();
```

Listagem I.11: Escolha do tipo do sensor DHT

```
1   // Uncomment whatever type you're using!
2   #define DHTTYPE DHT11   // DHT 11
3   //#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
4   //#define DHTTYPE DHT21   // DHT 21 (AM2301)
```

Listagem I.12: Funções do sensor DHT11

```
1   // Reading temperature or humidity takes about 250 milliseconds!
2     // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
3     float h = dht.readHumidity();
4     // Read temperature as Celsius (the default)
5     float t = dht.readTemperature();
6     // Read temperature as Fahrenheit (isFahrenheit = true)
7     float f = dht.readTemperature(true);
```

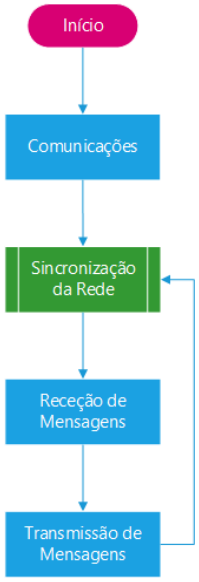

Figura I.14: Representação geral do programa MasterV1.ino por um fluxograma

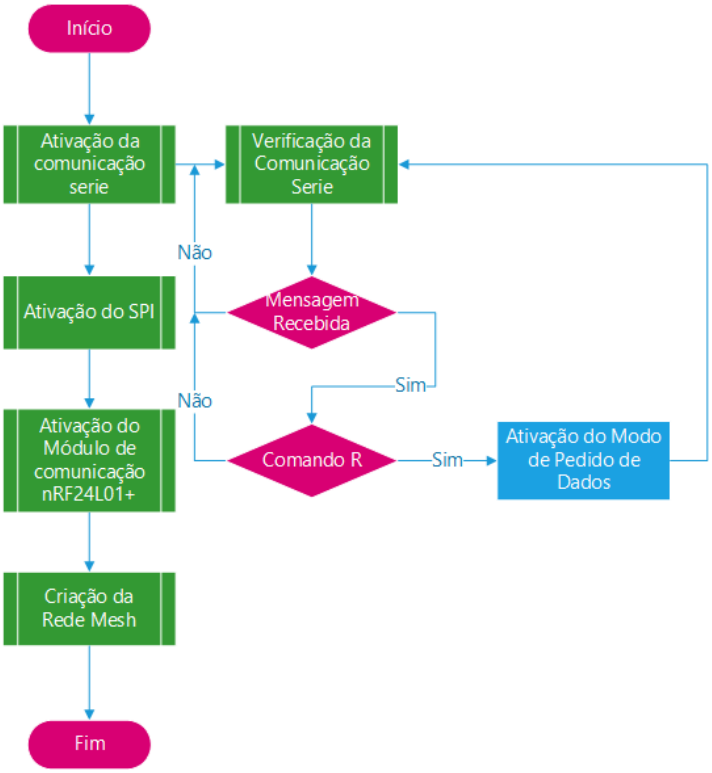

Figura I.15: Representação do processo de comunicações do programa MasterV1.ino por um fluxograma

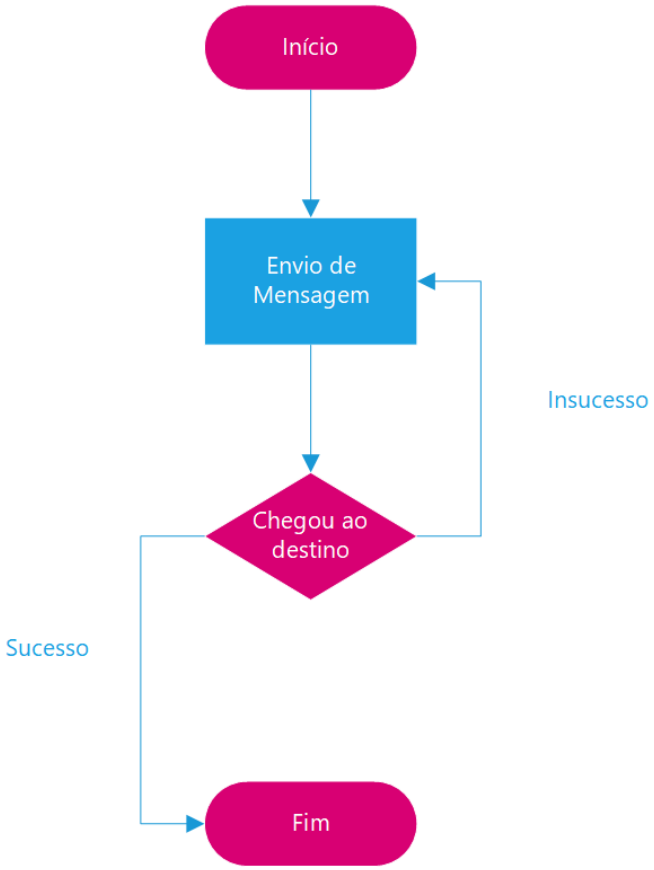

Figura I.16: Representação do processo de transmissão de mensagens do programa MasterV1.ino por um fluxograma

```
8
9   // Check if any reads failed and exit early (to try again).
10  if (isnan(h) || isnan(t) || isnan(f)) {
11    Serial.println("Failed to read from DHT sensor!");
12    return;
13  }
14
15  // Compute heat index in Fahrenheit (the default)
16  float hif = dht.computeHeatIndex(f, h);
17  // Compute heat index in Celsius (isFahreheit = false)
18  float hic = dht.computeHeatIndex(t, h, false);
```

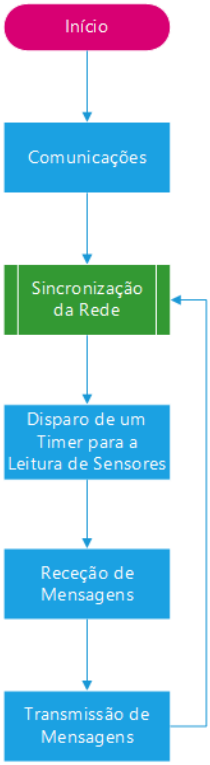

Figura I.17: Representação geral do programa SlaveV1.ino por um fluxograma

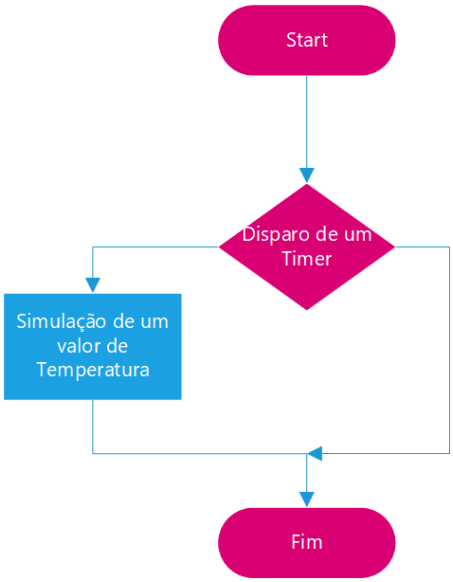

Figura I.18: Representação do processo Timer do programa SlaveV1.ino por um fluxograma

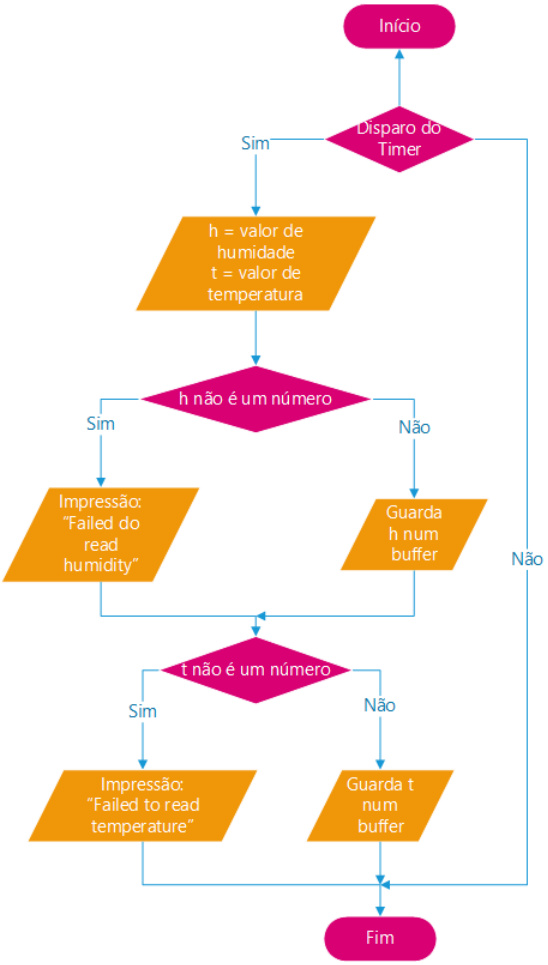

Figura I.19: Representação do processo Timer no programa "SlaveV2.ino

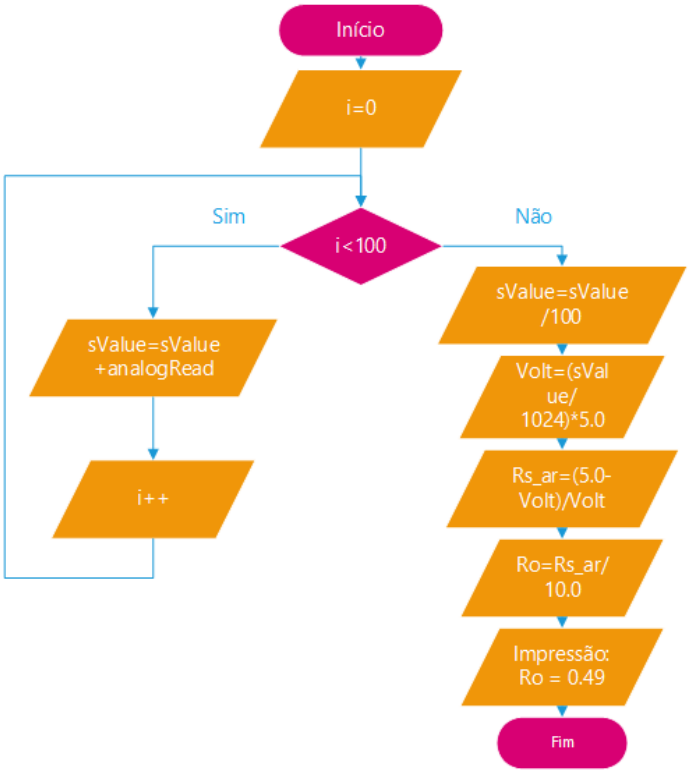

Figura I.20: Representação do programa ”MQ2_getValues.ino"por um fluxograma

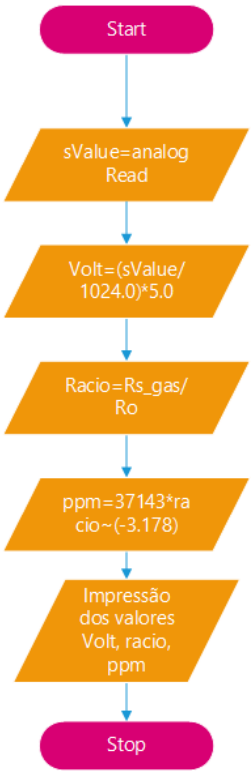

Figura I.21: Representação do programa "MQ2_gas_detection.ino"por um fluxograma

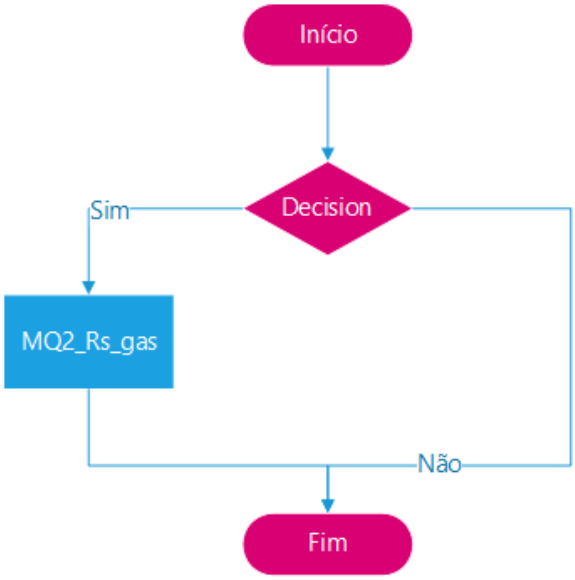

Figura I.22: Representação do processo Timer no programa "SlaveV2.ino"por um fluxograma

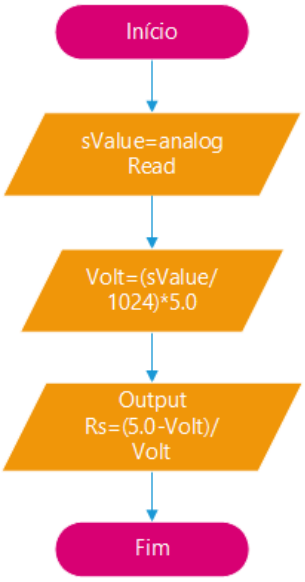

Figura I.23: Representação da função "$MQ2_RS_gas()$"por um fluxograma

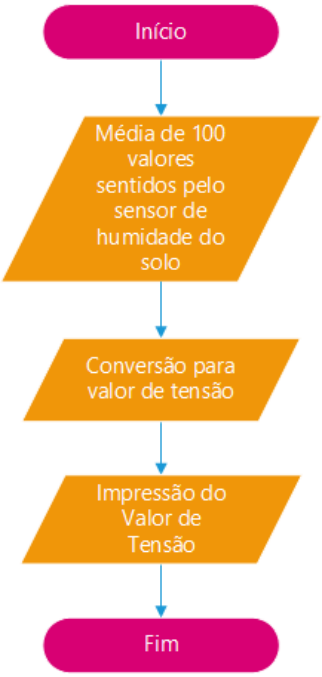

Figura I.24: Representação do programa "Soil_Moisture_GetDataValues.ino"por um fluxograma

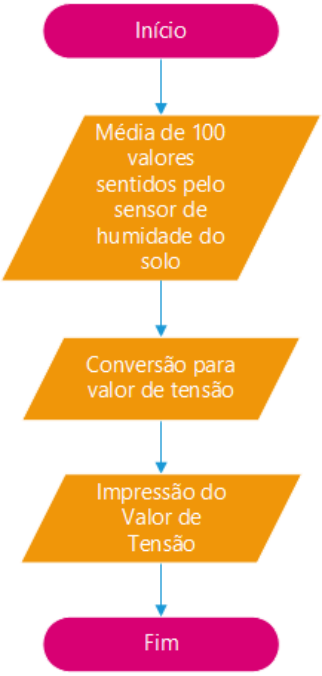

Figura I.25: Representação do programa "Soil_Moisture_getDataValues.ino"por um fluxograma

I.9 Resultados

I.9.1 Node/Hop 1

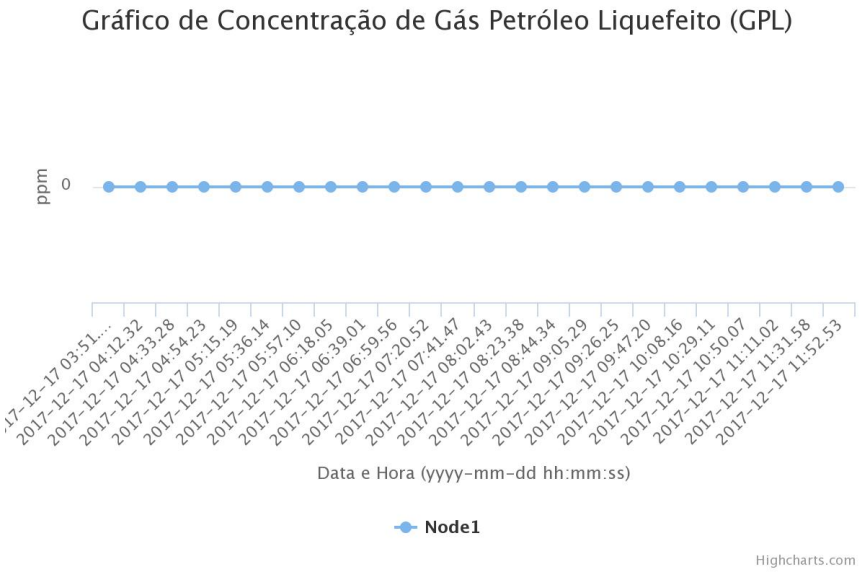

Figura I.26: Resultado obtido na leitura do sensor MQ2 - GPL no Node 1

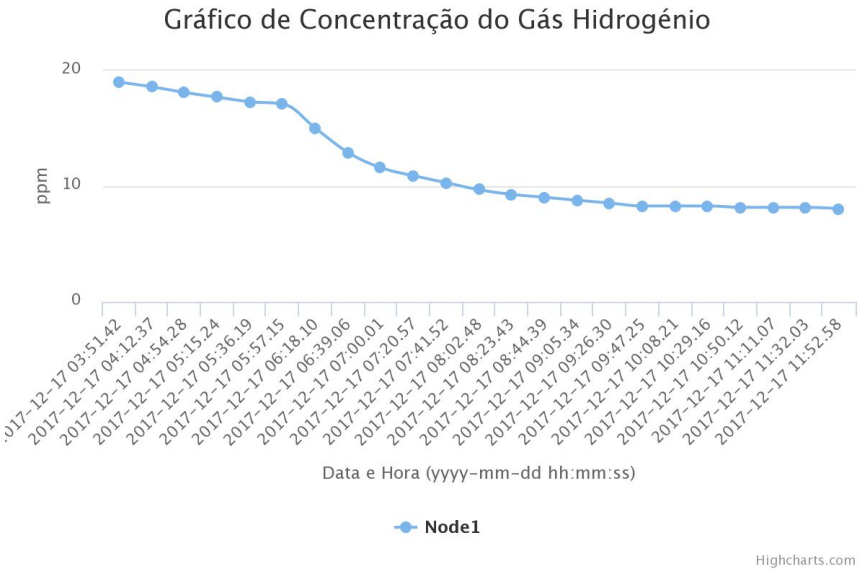

Figura I.27: Resultado obtido na leitura do sensor MQ2 - Hidrogénio no Node 1

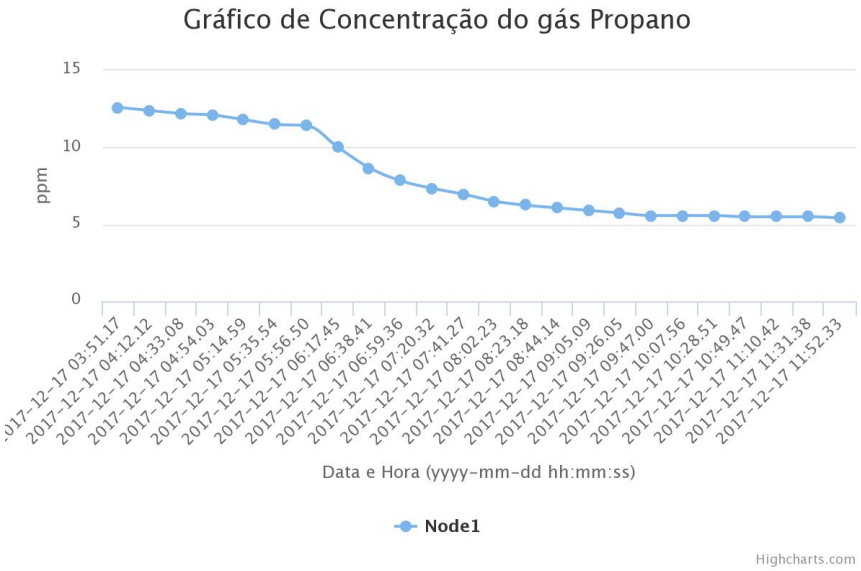

Figura I.28: Resultado obtido na leitura do sensor MQ2 - Propano no Node 1

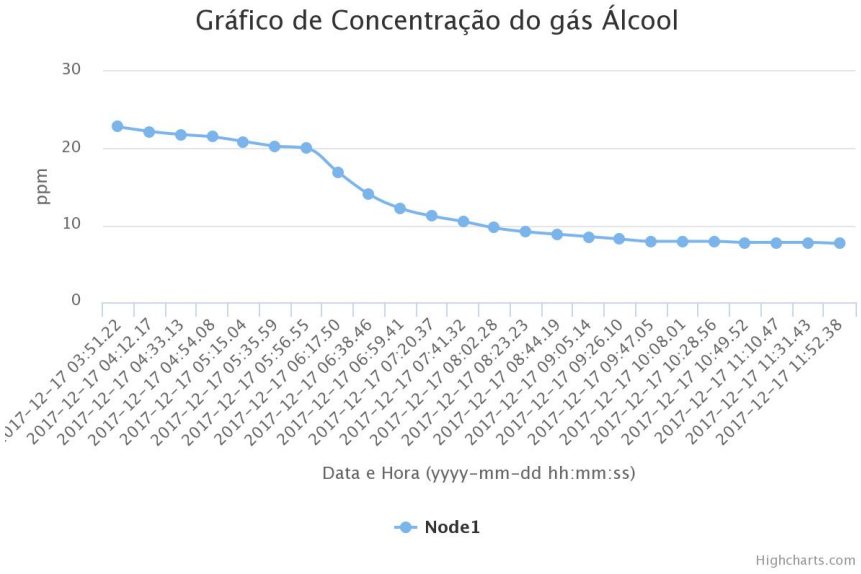

Figura I.29: Resultado obtido na leitura do sensor MQ2 - Alcool no Node 1

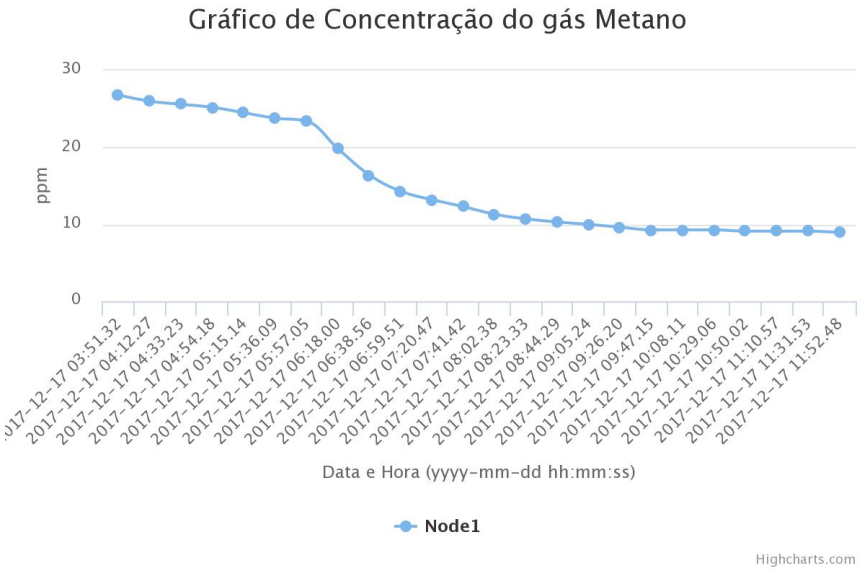

Figura I.30: Resultado obtido na leitura do sensor MQ2 - Metano no Node 1

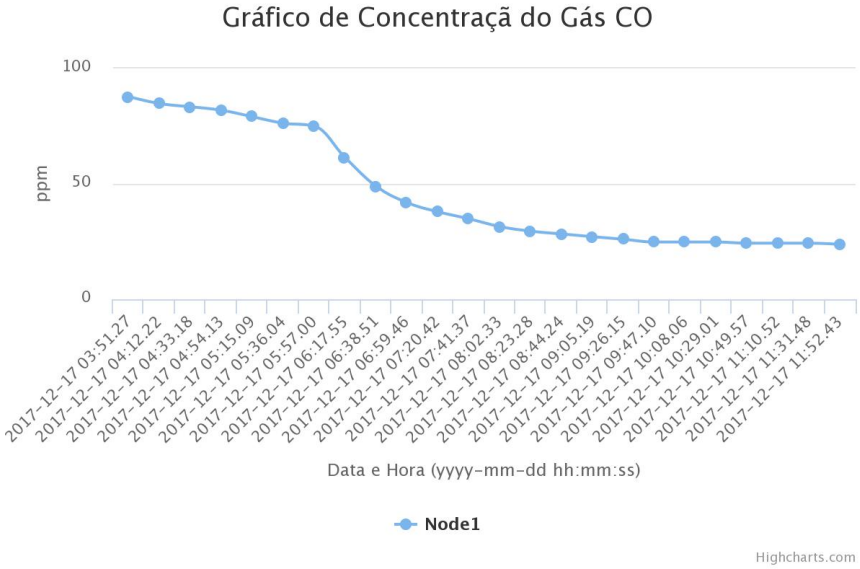

Figura I.31: Resultado obtido na leitura do sensor MQ2 - Monóxido de Carbono no Node 1

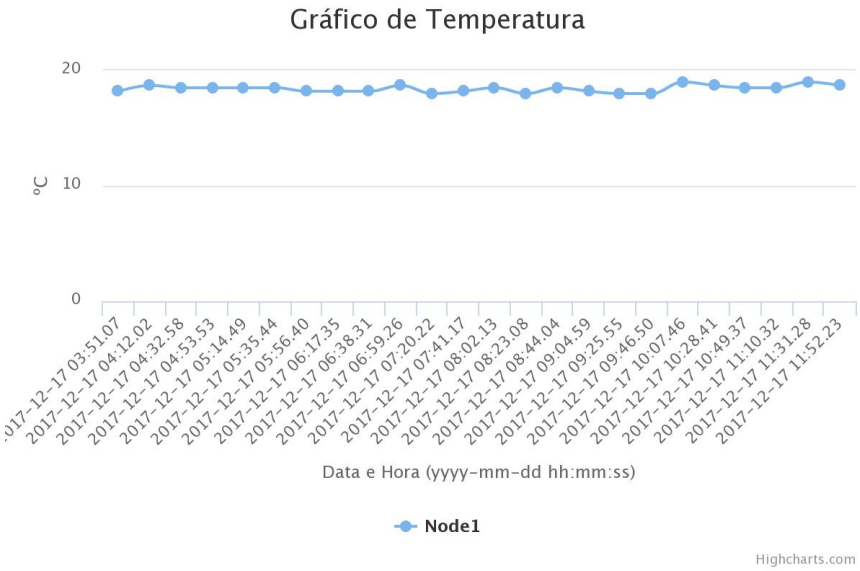

Figura I.32: Resultado obtido na leitura do sensor TMP36 e Max38155 - Temperatura no Node 1

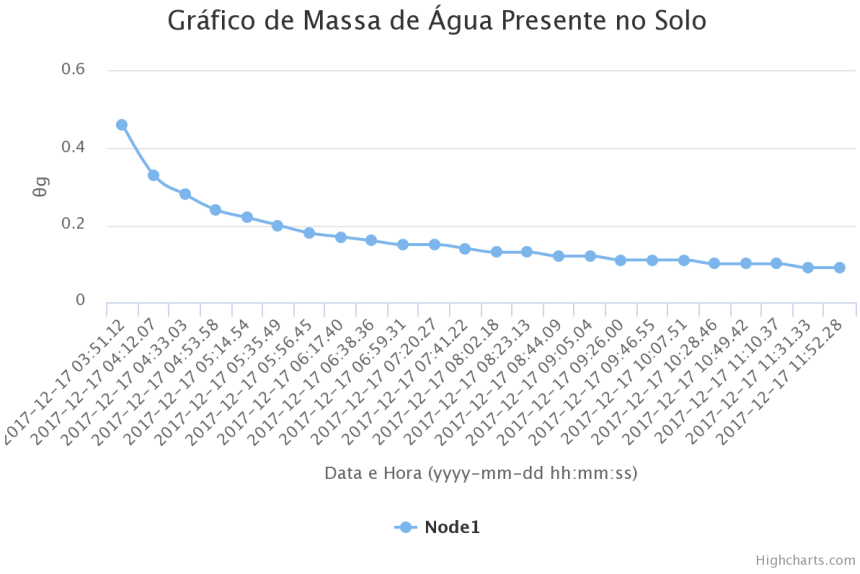

Figura I.33: Resultado obtido na leitura do sensor Humidade do Solo - Humidade do Solo no Node 1

I.9.2 Node/Hop 2

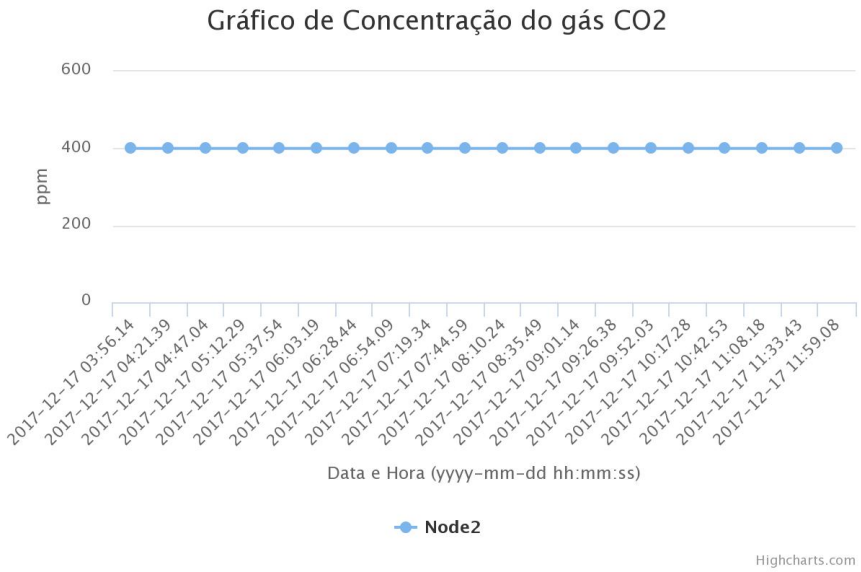

Figura I.34: Resultado obtido na leitura do sensor MG811 - CO2 no Node 2

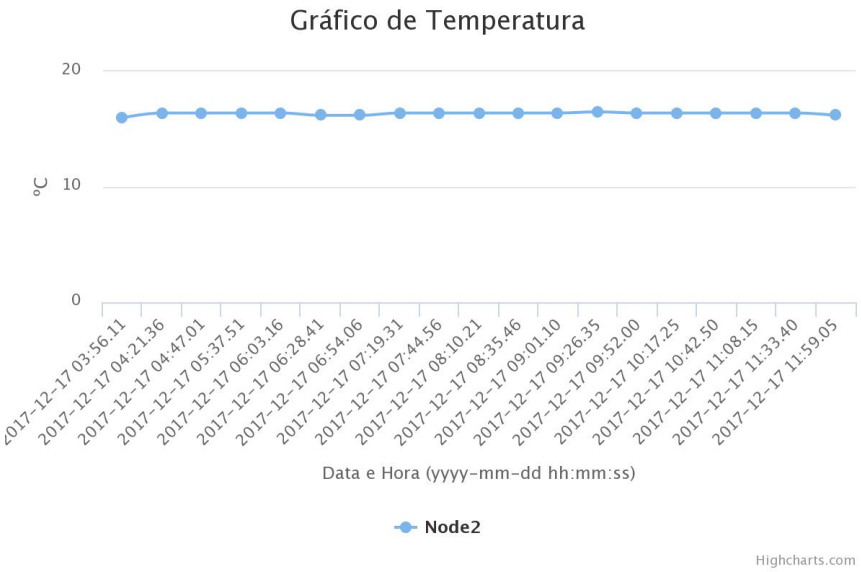

Figura I.35: Resultado obtido na leitura do sensor MPL115A2 - CO2 no Node 2

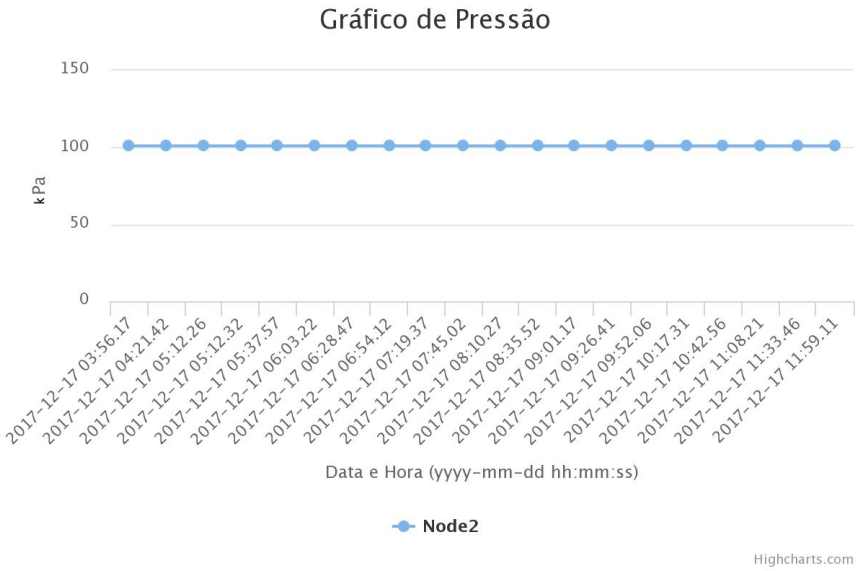

Figura I.36: Resultado obtido na leitura do sensor MPL115A2 - Pressão no Node 1